Naval Research Laboratory
Washington, DC 20375-5320

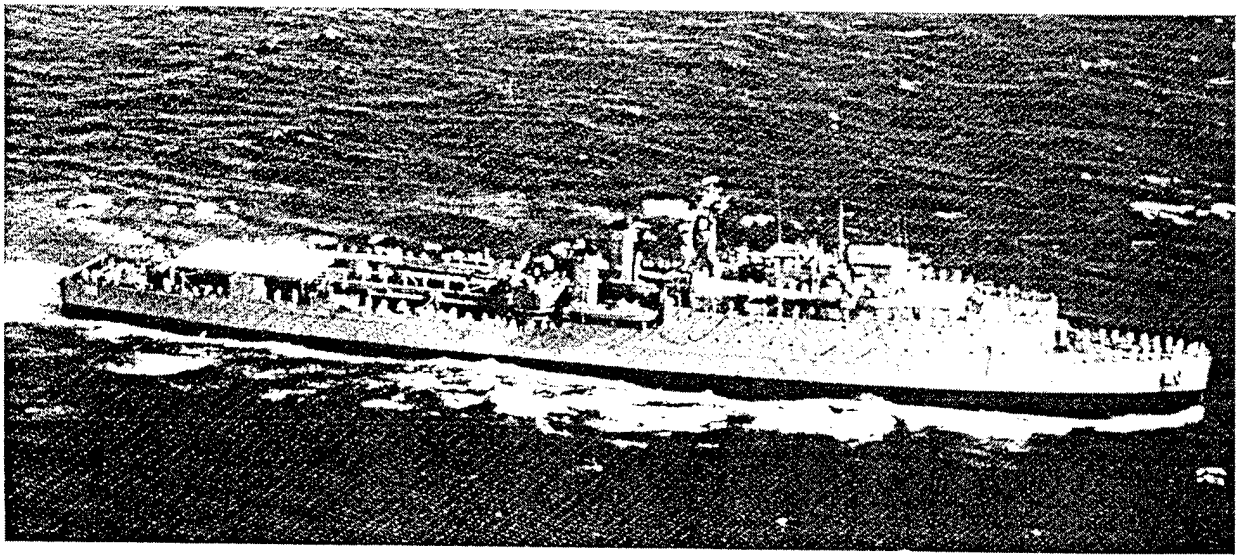
NRL/MR/6180--01-8559

Supervisory Control System for Ship Damage Control: Volume 1 — Design Overview

David C. Wilkins
Janet A. Sniezek
Beckman Institute
University of Illinois, Urbana, Illinois

Patricia A. Tatem
Frederick W. Williams
Navy Technology Center for Safety and Survivability
Chemistry Division

July 27, 2001



REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave Blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	July 27, 2001	Final FY 1999-FY 2001

4. TITLE AND SUBTITLE

Supervisory Control System for Ship Damage Control: Volume 1 — Design Overview

5. FUNDING NUMBERS

PE - 63508N

6. AUTHOR(S)

D.C. Wilkins,* J.A. Sniezek,* P.A. Tatem, and F.W. Williams

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Research Laboratory, Code 6180
4555 Overlook Avenue, SW
Washington, DC 20375-5320

8. PERFORMING ORGANIZATION REPORT NUMBER

NRL/MR/6180--01-8559

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Naval Research
800 North Quincy Street
Arlington, VA 22217-5660

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

*Beckman Institute, University of Illinois, Urbana IL 61801

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

This report presents a state-of-the-art concept of automated situation awareness for ship damage control. The solution encompasses model-based crisis recognition, model-based predictive validation, automated casualty response, and a supervisor interface console.

14. SUBJECT TERMS

Flooding
Fire
Damage control
Supervisory control
Automation

15. NUMBER OF PAGES

111

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std 239-18
298-102

CONTENTS

SUPERVISORY CONTROL SYSTEM FOR SHIP DAMAGE CONTROL:

VOLUME 1 – DESIGN OVERVIEW

1. Forward and Acknowledgments

This report provides a detailed design of a "Damage Control -- Supervisory Control System" (DC-SCS) for ship damage control. A version of this report was first distributed to the Naval Research Laboratory on October 23, 1996; the technical content of this current report is the same except for cosmetic changes. The overall design and design rationale described in this report have not been superseded in the last four years, and we continue to use this report as the guiding light for the realization of a supervisory control system for ship damage control that is being created as part of the "Damage Control – Automation for Reduced Manning" (DC-ARM) program of the Naval Research Laboratory.

Between September 1997 and October 2000, two research groups at the University of Illinois have actively pursued the realization of the design described herein -- the Knowledge Based System Group located in the Beckman Institute for Advanced Science and Technology and directed by Professor David Wilkins, and the Judgment and Decision Making Group located in the Psychology Department and directed by Professor Janet Sniezek. Additional technical reports describing much of the progress during the three-year period from 1997-2000 are described in a series of supplementary volumes to this report that are being published concurrently. The titles of these volumes are as follows:

- Volume 1: Design Overview
- Volume 2: Scenario Generation and Physical Ship Simulation
- Volume 3: Human Computer Interaction and Visualization
- Volume 4: Intelligent Reasoning
- Volume 5: Knowledge Ontology
- Volume 6: Experimental Measurement of Situation Assessment
- Volume 7: Software Architecture
- Volume 8: User's Manual
- Volume 9: Digital ex-USS *Shadwell* Representation

This DC-SCS design described herein and the corresponding components of the research prototype that was completed in 1996 owes heavily to many people. Special thanks go to Vadim Bulitko, Eugene Grois, William Hsu, and Surya Ramchandran, who are the primary architects of the described Supervisory Control system for the ship damage control domain, and who contributed important sections of this report.

Michael Baumann, Michele Donovan, and Lyn Van Swol made major contributions to the Supervisor-Operator Interface section of this report. With respect to the design and

Manuscript approved May 30, 2001.

implementation of the Physical Ship Simulation and Visualization Subsystems, we are indebted to Peter Baer, Scott Borton, Ron Carbonari, Brian Katz, David Kruse, Aaron Levinson, Eric Lin, Arthur Menaker, Guoming Shou, Yi Su, Marty Weller, and James Young. With respect to the design and implementation of the Intelligent Reasoning and Total Ship Representation Subsystems, we are indebted to Kees Cook, Gontran Duchesne, Carl Fagerlin, Vicki Fairchild, Luca Faggioli, Kurt Gimbel, Anthony Haynes, Vladimir Jojic, Marcia Leal, Ole Mengshoel, John Viene, Jerry Schlabach, Brent Spillner, Baogang Yao, and Pavel Yusim. Michael Daniels, Daniel Maser, Casey Ryan, Josh Sherman, and Naxin Zheng provided computer system administration support. Marcia Snow, Audrey Fisher, Jamie Carras and Nathan Otis provided recent reformatting of this report.

This research effort would not have been possible without the expertise provided to us by experts in the domain of ship damage control. We gratefully acknowledge the help provided by Eric Allely, Rik Breaux, Cliff Campbell, Cmdr. Jose Cicneros, John Dietish, Cmdr. Timothy Steele, David Tate, Patricia Tatem, Frederick Williams, Bernie Ulozas and Herb Wolk.

The DC-SCS design described in this report is a generalization of the MINERVA intelligent reasoning shell, and the DC-TRAIN damage control simulator and immersive trainer. Funding for these two efforts were provided by grants from the Office of Naval Research: N00014-88K-0124, N00014-94-1-0432, N00014-95-1-0934, and N00014-95-1-0749; and by a grant from the Air Force Office of Scientific Research: Grant F49260-92-J-0545.

2. Introduction

The report details a solution approach to the Supervisory Control decision-making task for damage control management of fire, smoke, flooding, pipe rupture, and stability aboard ships. The solution is relevant to ships built in the future, as it assumes intelligent sensors and actuators that are more advanced than those that exist on any current ship. The solution encompasses all three stages of Supervisory Control: model-based crisis recognition, model-based predictive validation, and automated casualty response. This report also describes an approach to the creation and evaluation of a Supervisor-Operator Override Console because of its integral relationship to Situation Assessment. The described Supervisory Control solution is designed to function ***in vitro***, by use of the existing Illinois Damage Control Scenario Generator armed with simulated smart sensors and actuators. And it is designed to function ***in vivo***, in live crisis exercises aboard a floating research laboratory called the ex-USS *Shadwell*. The three major research areas of the described Supervisory Control solution are knowledge-based expert systems, machine learning in noisy domains, and assessment of supervisory human-computer interfaces.

Five innovative aspects of the solution approach described in this report are as follows. *First*, with respect to the creation of the Crisis Recognition System, the solution uses the Illinois Damage Control Scenario Generator (which simulates primary and secondary damage of fire, smoke, flooding, and rupture) to generate hundreds of thousands of unique training examples. These examples are used to train neural network and symbolic machine learning algorithms to recognize novel crisis situations and not be misled by unusual ship states. *Second*, with respect to the Predictive Validation System, the solution uses the Illinois Damage Control Scenario Generator for model-based prediction to rapidly verify suspected crisis situations. *Third*, to

manage the complexity of the Supervisory Control task, our solution decomposes the overall Supervisory Control task into 19 independent modules that communicate solely with an SQL relational database, and that can be independently developed and tested. Currently, we have demonstrable initial working versions of 12 of the 19 Supervisory Control modules. *Fourth,* with respect to the Supervisor Supervisory Control and Override Console, our solution involves conducting extensive experiments with seasoned damage control assistants to assess key aspects of the Supervisory Interface: the degree to which experts have confidence in the automated system, the degree to which the system communicates its situation awareness understanding to the experts, and the degree to which experts successfully override its decisions. *Fifth,* to accelerate advances in model-based fire spread, which in the long term is *essential* for accurate predictive validation, our solution involves the creation of a test bed for advanced combustibility research for use by various research groups around the country who have an interest in the refinement of the CFAST fire and smoke propagation algorithm for ship-specific structures, such as air tight compartment structures and ship ventilation shafts.

The described solution to the Supervisory Control task addresses the following challenges:

- How will real-time recognition models be used to detect crises?
- How will predictive models be used to validate the existence of a crisis?
- How will the knowledge base be refined so as to deal with false positive and negative sensor readings and other sources of noise?
- How will a knowledge base be constructed that can recognize novel crisis situations that haven't been explicitly programmed for?
- How will existing static and dynamic databases be integrated into the Supervisory Control reasoning process?
- How will the Supervisory Control model be evaluated? How can the process of solving the task be decomposed so that strong indicators of progress or its lack are ascertained at regular intervals?
- How can situational awareness and problem solving be communicated to the supervisor to prevent errors in supervisory control decisions?
- Can the solution algorithms be made to operate from first principles, so that a solution system is not dependent on aspects of the ship description that is used during development?
- How can the solution system be made to run in real-time?
- How can CFAST, which is a key component of any model-based crisis prediction method, be made to run over all compartments of a ship in real-time?

The remainder of the report is organized as follows. Section 3 systematically describes our solution approach for automated Supervisory Control and a Supervisor-Operator Override Console. It begins with a high-level overview, and then proceeds to a subsystem level and module level description. Finally, it gives a description of human data collection and performance evaluation.

Section 3.1 provides four different high-level overview perspectives of our entire Supervisory Control System. The goal of this section is to provide the reader with a good initial orientation to our proposed solution.

Section 3.2 describes the six modules of the Human-Ship Interface Subsystem, which include GUI interfaces for ship specification, scenario specification, and Supervisor Control. It also covers interfaces to ship sensors and actuators, and a Web interface that would allow all GUI interfaces to be accessed over the Web.

Section 3.3 describes the five modules of the Ship Simulation Subsystem, which is responsible for primary and secondary damage simulation of fire, smoke, flooding, rupture, and stability.

Section 3.4 describes the three modules of the Knowledge Base Subsystem, which contains the domain-specific knowledge used by all parts of the Supervisory Control system, ranging from graphic visualization, numerical ship simulation, to intelligent reasoning.

Section 3.5 describes the five modules of the Intelligent Reasoning Subsystem, which is responsible for non-numerical reasoning needed for the Supervisory Control Task. These include crisis recognition reasoning, casualty response reasoning, predictive crisis validation, machine learning, and simulated intelligent ship objects.

Section 3.6 describes the importance of the Supervisory Interface along with a data collection methodology to ensure that the Navy damage control community has confidence in all aspects of its operation.

Section 3.7 describes evaluation and testing.

Section 3.8 describes the various interdependencies among system modules.

After presenting our proposed solution in Section 3, the report continues with Section 4, which covers progress to date. Section 5 details related research that is relevant to the Supervisory Control task. Finally, Section 6-8 provides a summary and references.

3. Solution Approach

3.1 Four Solution Overviews

The design architecture of our system for automated Supervisory Control for the ship damage control domain is complex. Therefore, Section 3.1 provides a number of snapshots of the way that the overall system works.

3.1.1 Solution Overview 1 of 4: High-Level Functional Overview

In this section, we describe a novel multi-blackboard based expert system capable of dealing with the automated Supervisory Control challenges imposed by the damage control domain. A high-level structure of the system is depicted in Figure 1. The major objects shown in the Figure 1 functional overview are as follows:

Supervisory Interface. Although our automated Supervisory Control System is capable of functioning without human intervention, it is still important to provide ship personnel with the ability to override the system. The ability to properly override the system depends on high-quality knowledge, which means that the Supervisor Operator interface must provide a quick view of the current state of the ship, the readings of sensors, and the conclusions that the automated system has reached regarding the best and alternative courses of decisions.

Crisis Recognition Expert System. The crisis recognition system has the responsibility of monitoring the state of the ship and detecting abnormal states that require damage control response.

Casualty Response Expert System. The casualty response system has the responsibility of determining the best course of action to address a damage control problem that involves fire, smoke, flooding, rupture, or stability. It sends commands to actuators that carry out the casualty response.

Illinois Damage Control Scenario Simulator. The Illinois Damage Control Scenario Generator simulates primary and secondary damage of fire, smoke, flooding, rupture, and stability. It is used to generate classified training learning examples for the neural and symbolic learning algorithms. It is used by the Predictive Crisis Validation System to verify the existence of a situation that requires damage control response. The system simulates objects on a ship, such as the fire main, chilled water, and other ship systems relevant to damage control.

Learning Modules. The learning modules assist with knowledge base refinement of the crisis recognition and casualty response expert systems. They use different techniques for different knowledge representations. For the parts of the expert systems that use symbolic knowledge, rule-learning algorithms are used. For the parts of the expert systems that are based on neural networks, neural network learning algorithms are used. For the parts of

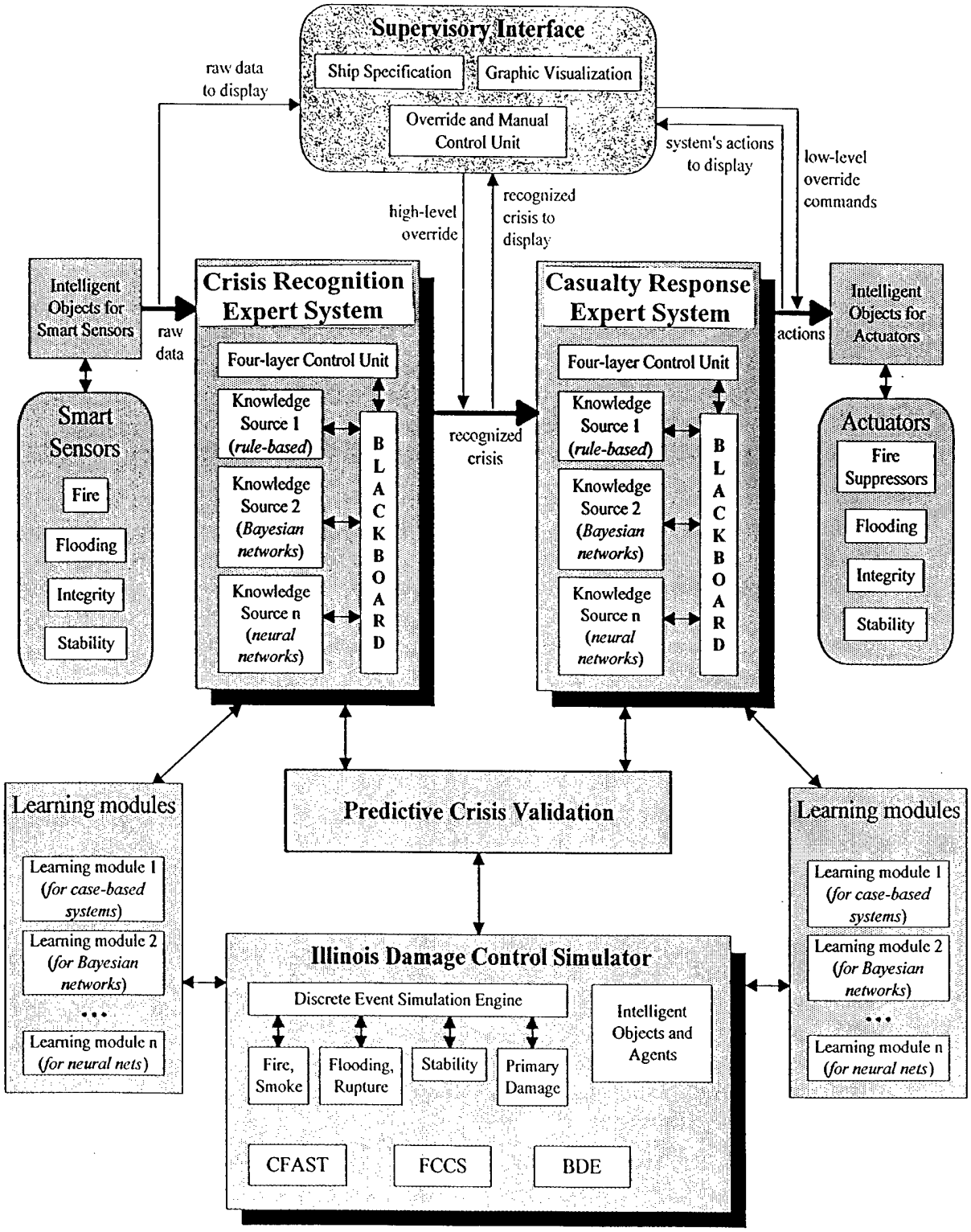


Figure 1. Functional overview of the Automated Supervisory Control System

the expert systems based on case-based reasoning or belief networks, the appropriate learning algorithms are used for these types of knowledge representations.

Predictive Crisis Validation. Predictive validation operates using several of the other systems. It is initially called into action by the crisis recognition system, uses the values that are generated by the smart sensors, and uses the Illinois Damage Control Scenario Simulator.

Smart sensors and Smart Actuator. In addition to interfacing to actual smart sensors and actuators, Figure 1 shows Intelligent Objects for sensors and actuators. These intelligent objects simulate the behavior of sensor and actuator hardware. This software simulation allows ***in vitro*** damage control experiments to be conducted using the Illinois Damage Control Scenario Generator. These ***in vitro*** experiments are essential for neural network learning and refinement of the crisis recognition and casualty response knowledge bases.

3.1.2 Solution Overview 2 of 4: PC-Level Overview

Our solution is decomposed into the four major subsystems shown in Figure 2. This decomposition has a number of advantages ranging from computational efficiency to clarity of organization. The four major subsystems shown in Figure 2 are as follows:

Human-Ship Interfaces: Visualizes the ship status and the status of the Supervisory Control expert system for the supervisor. Interfaces to intelligent objects representing smart sensors and actuators.

Ship Crisis Simulator: Numerical simulation of crisis progression and the effect of damage control action on the ship, in the areas of fire, smoke, flooding, pipe rupture, and stability. The ship simulator is necessary for predictive validation and machine learning.

Total Ship Representation: Uses a relational database representation for all static and dynamic domain knowledge, including ship visualization, numerical ship simulation, and intelligent reasoning. The other PCs access this information over a 100 Mbps local area network. For purposes of efficiency, we are also considering database replication and synchronization, whereby the Total Ship Representation would reside on more than one PC.

Intelligent Reasoning System: Performs crisis recognition, predictive validation, neural network and symbolic learning, and simulation of intelligent objects.

Specific examples of how the above four-subsystems break down follows. Consider a particular fire main pipe in a particular compartment of the ship. All knowledge concerning the pipe, such as its dimensions, composition, thickness, susceptibility to damage from vibration, visualization, and current pressure are stored in the total ship representation subsystem. The ship Crisis simulator subsystem contains the numerical algorithms that determine the pressure inside the fire main at any point of time, and the rate of flow of a rupture of the fire main in the case of damage. The intelligent reasoning system contains the programs that reason about this fire main pipe fragment. It would reason topologically on how the fire main cutoff valves and fire pumps should be altered in the case of a specific rupture in the fire main system.

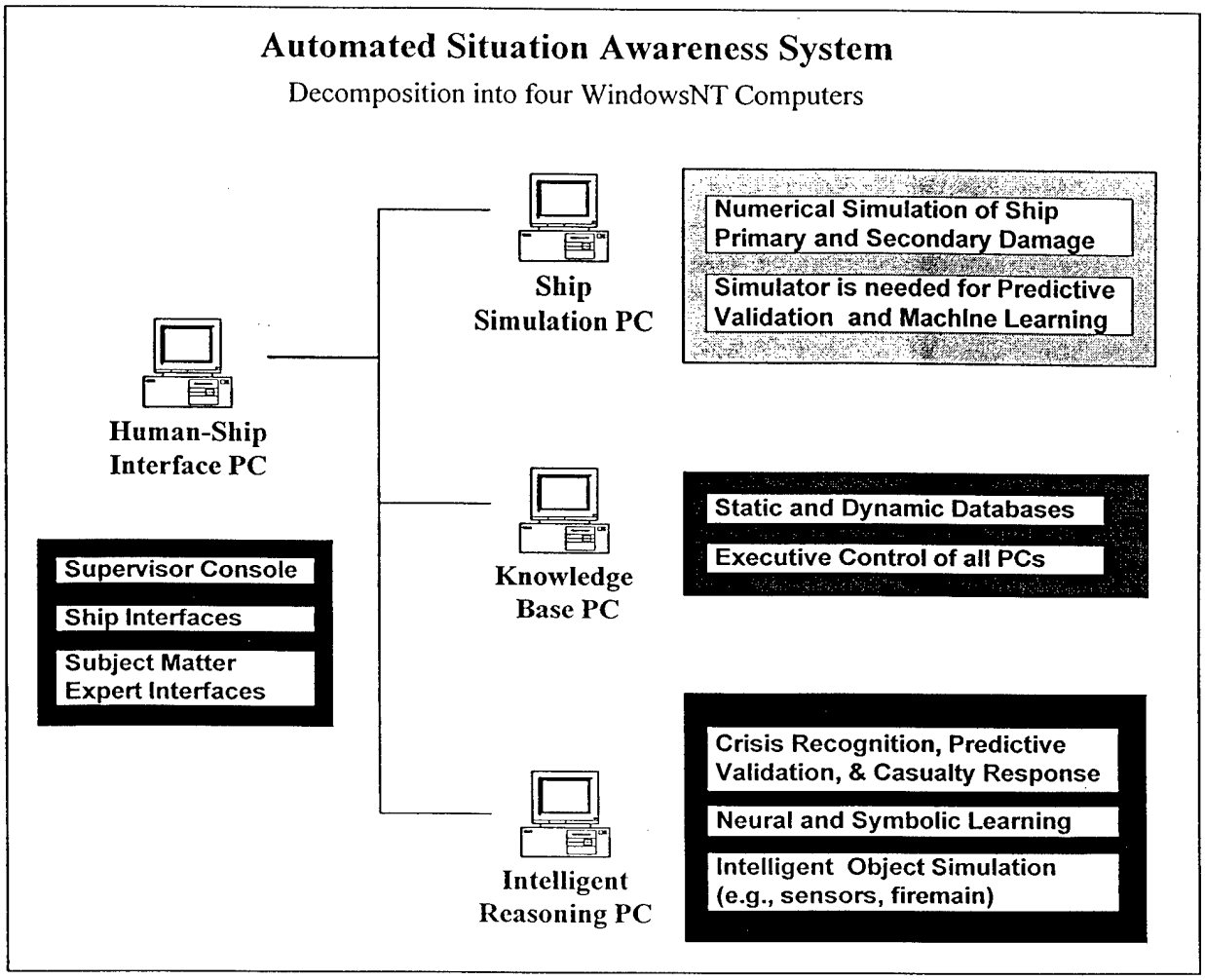


Figure 2. PC-Level View: Decomposition of the Automated Supervisory Control System into 4-networked computers

3.1.3 Solution Overview 3 of 4: Module-Level Overview

The lowest level of organization of the Supervisory Control System is the module-level and this is shown in Figure 3. An ironclad rule is that a module interfaces only with the Structured Query Language (SQL) Database Module through an Open Data Base Conductivity (ODBC) interface. This greatly simplifies the design, development, and modular testing of the overall system. A module can be understood or changed without the need for detailed knowledge of the internal implementation of any other module in the system. Modules are allowed to re-represent knowledge in the data structures of their choice for purposes of efficiency. But they must always initially read in this knowledge from the SQL database module. Whenever a module makes an inference relevant to any other module in the Supervisory Control system, it sends this knowledge into the SQL Database module. The 19 modules that comprise the system are examined in detail later in the current section. For now, we simply list them:

Subsystem #1: Human-Computer and Ship Interface Subsystem

- Scenario Specification Module
- Supervisory Interface Module
- Ship Specification Module
- Ship Visualization Module
- Web Interface Module
- Ship Sensor and Actuator Interface Module

Subsystem #2: Physical Ship Simulation Subsystem

- Primary Damage Module
- Fire and Smoke
- Flooding and Rupture
- Stability
- Discrete Event Simulation Module

Subsystem #3: Total Ship Representation Subsystem

- SQL Database Module
- Deductive and Temporal Inference Module
- Executive Control Module

Subsystem #4: Intelligent Reasoning Subsystem

- Opportunistic Blackboard
- Neural Networks
- Belief Networks
- Predictive Validation
- Intelligent Ship Objects

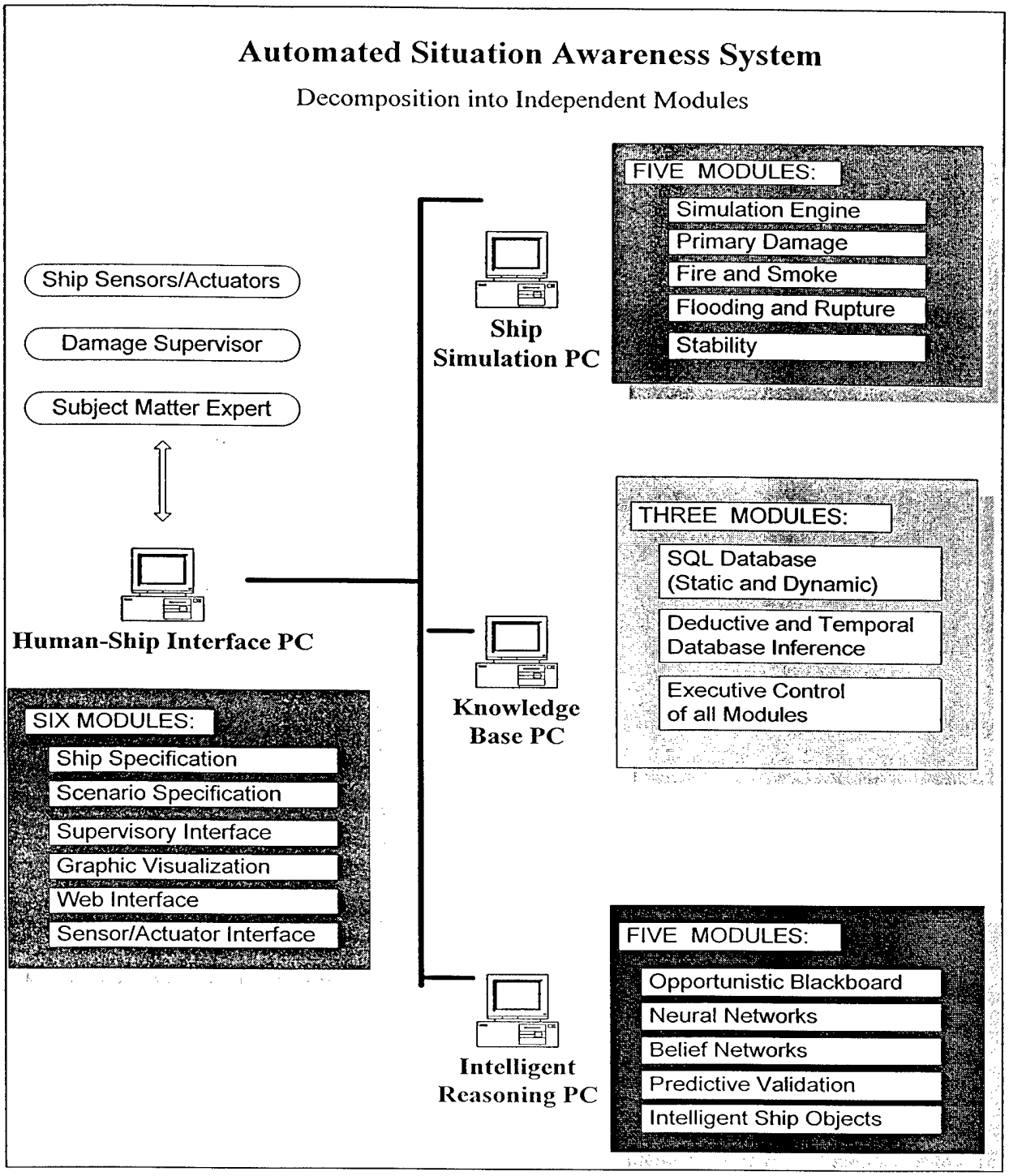


Figure 3. Module-Level View: Decomposition of the Automated Supervisory Control System into 19 independent modules

3.1.4 Solution Overview 4 of 4: In *Vitro* versus *In Vivo* Overview

Figure 4 shows the two main modes of operation that our Supervisory Control system can be operated in:

***In Vitro* Mode**: This mode uses simulated sensors and actuators. They mimic the actual sensors that are used and our ship simulator determines their values. This mode is required for neural network and symbolic learning of novel crisis situations. It has the further advantage of facilitating incremental evaluation of all elements of our solution prior to the deployment of our system aboard the ex-USS *Shadwell,* [*Williams, et al., 1987]*

***In Vivo* Mode:** The Supervisory Control system receives its input from real sensors aboard the ex-USS *Shadwell*, and activates real actuators. Actuator activation, in turn, changes the values that are read by the real sensors. The Illinois Damage Control Scenario Generator is still used, because it is required for predictive validation.

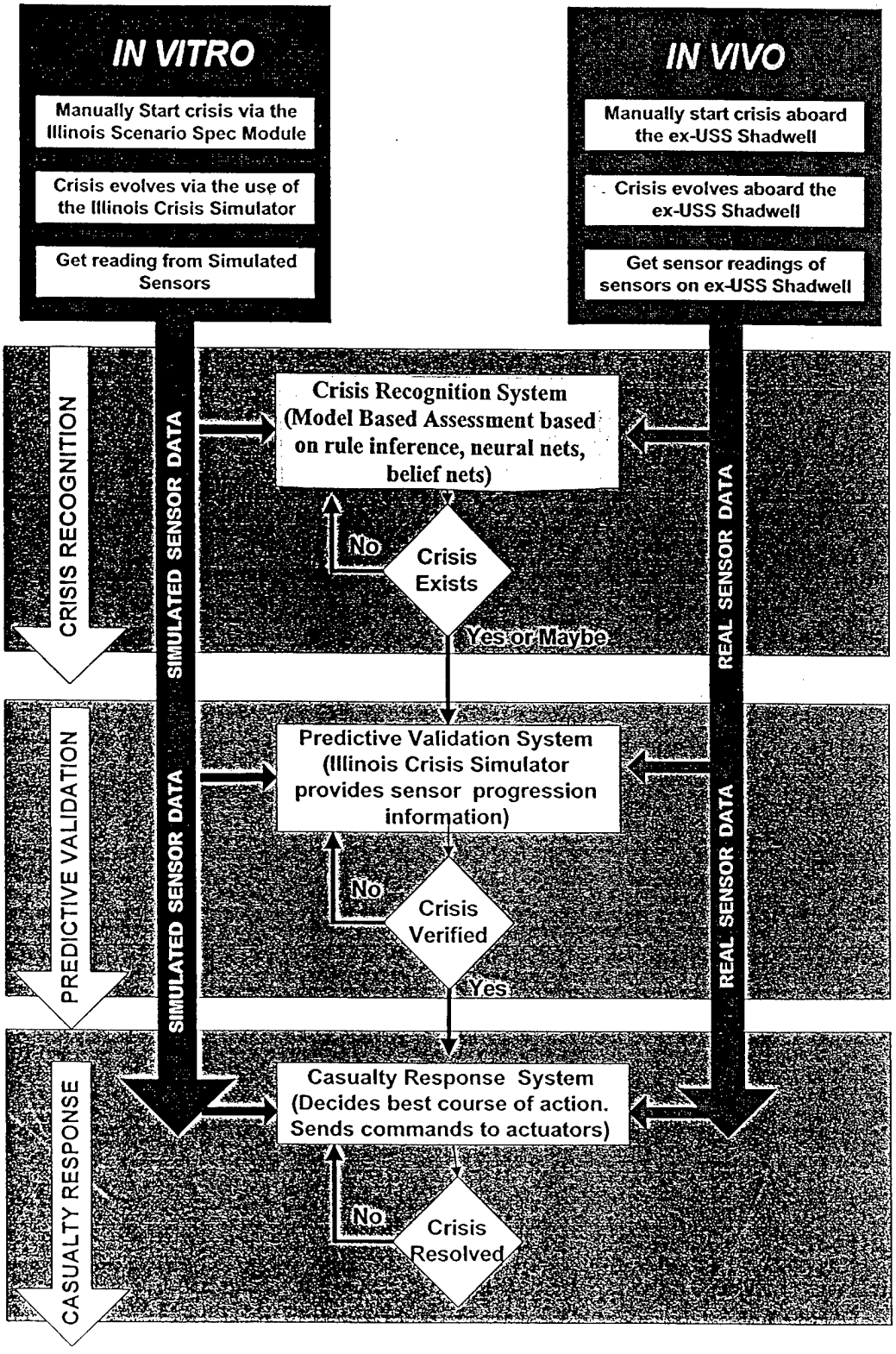


Figure 4. *In-Vitro* versus *In-Vivo* Automated Supervisory Control System

3.2 Subsystem 1: Human/Ship Interface Subsystem

The Human/Ship Interface subsystem provides indispensable links between the Supervisory Control system and a host of DC-ARM [Peatross, et al., 2001] users, including the damage control supervisor, subject matter experts, system developers and system evaluators; and it connects the Supervisory Control system to shipboard sensor and actuator systems. It provides an intuitive, informative and complete interface to all relevant Supervisory Control functions of DC-ARM. The human-computer interface (HCI) modules use proven graphical user interface methods and standards. The human/ship interface system provides essential project infrastructure, and the challenges relate more to producing good designs and developing robust code rather than solving Supervisory Control research problems. The subsections below describe each of the following six interface's modules:

- Ship Specification: subject matter experts can modify ship structure and contents
- Scenario Specification: scenario can be specified and executed as a what-if simulation
- Supervisor/Operator Interface: provides situation awareness; allows override control
- Ship Visualization: graphical method of providing situation assessment
- Web Interface: allows the above interfaces to run on any computer connected to Web
- Sensor/Actuator Interface: connects situation awareness system to physical ship

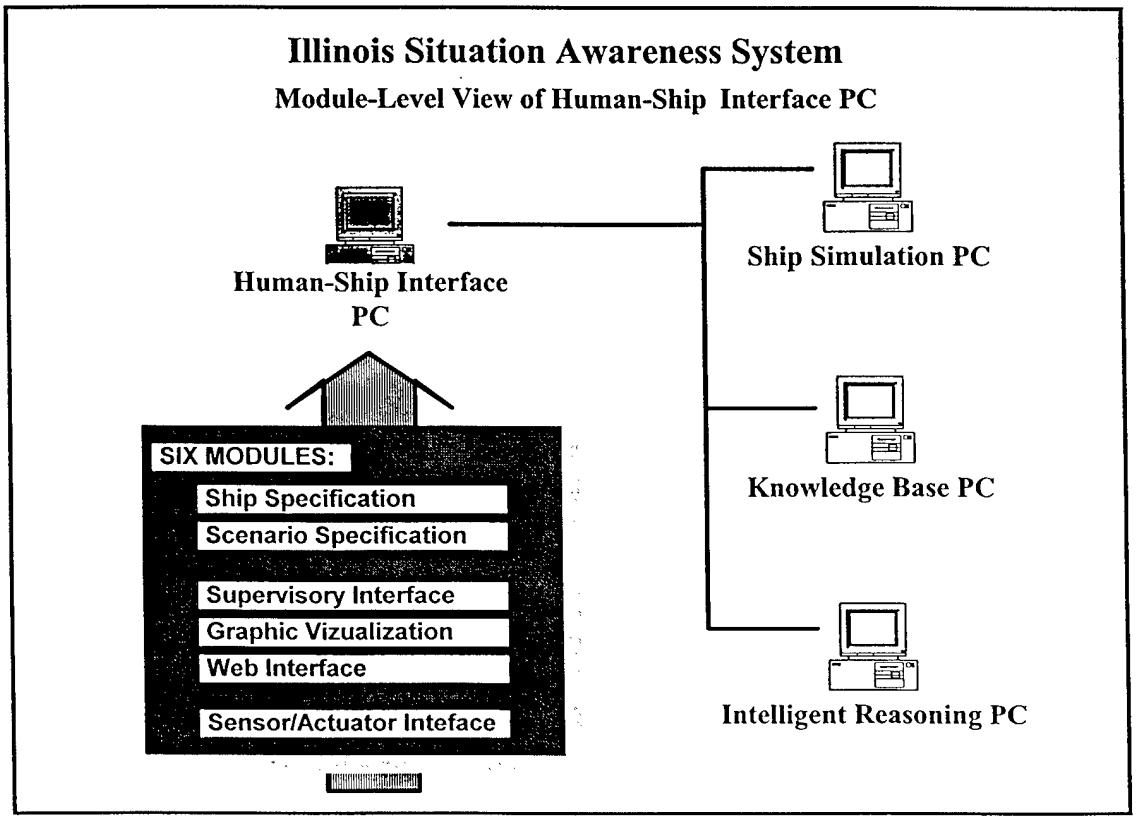


Figure 5. The six modules of the Human-Ship Interface PC

3.2.1 Ship Visualization Module

The graphical ship visualization module provides a 3-dimensional representation of the ship. The user can manipulate the view by using the menus, mouse and keyboard. The state of the ship is continuously updated by the situation awareness system. Events such as fire, smoke, flooding, pipe and vent status and other sensory data, as well as repair activity are displayed using color-coded compartments, icons and standard Navy notation (i.e., for repair activity). The user can rotate the view and zoom in for optimum viewing, and can optionally turn off the display of some items to reduce clutter. Additionally, it is possible to view the ship as a whole, split the decks to provide a representation similar to the one found on the damage control plates, or view decks individually. Figure 6 gives an example of our visualization system.

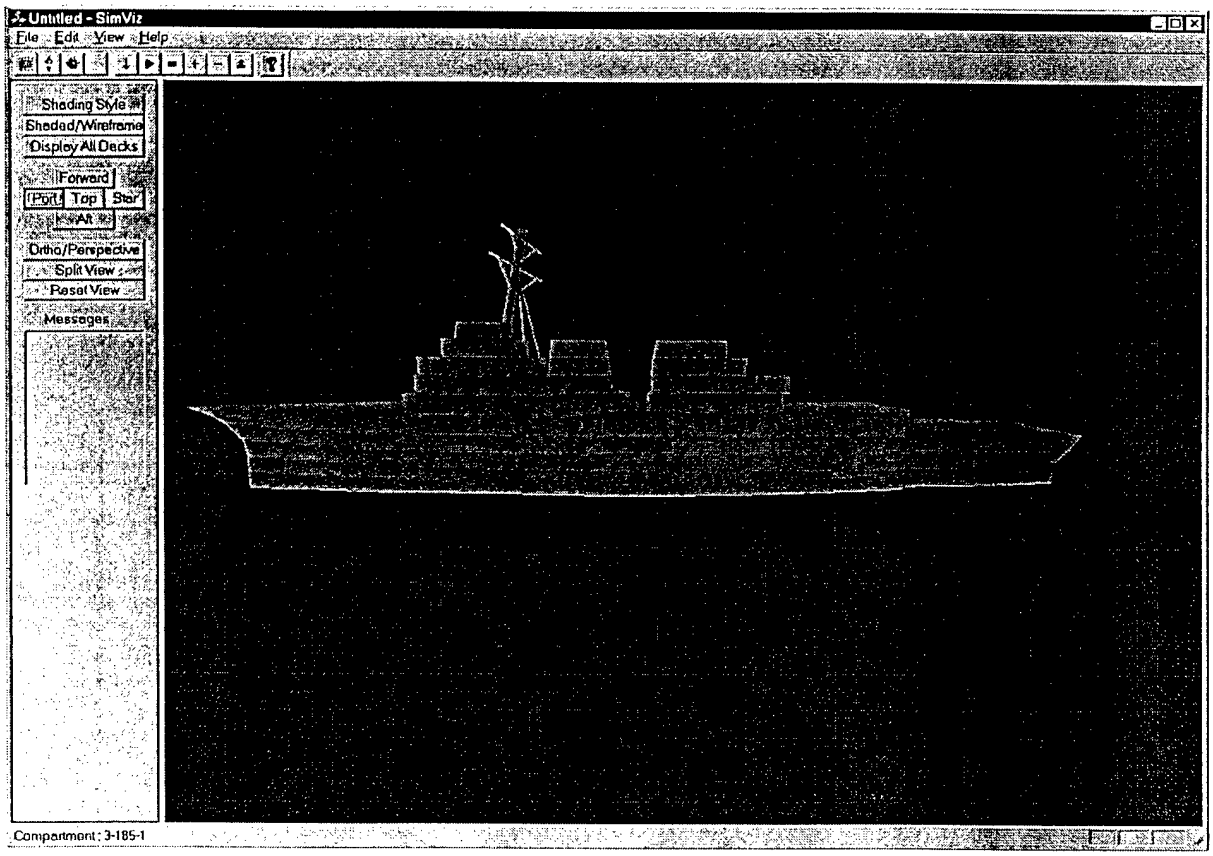


Figure 6. Illinois Ship Visualization System (figure 1 of 2): Port View. A more detailed view of the GUI toolbar interface on left is shown in the next figure

The graphical visualization module has many uses. The core interface and data are used in other modules such as the ship specification, scenario specification, and supervisor/operator override control. During a crisis, it provides the crisis supervisor with a clear, integrated view of the situation as an alternative to the DCS interface. We intend to allow the Supervisor to achieve situation awareness using either the DCS or the Illinois Graphic Visualization System. During the development and testing of the Supervisory Control system - ranging from crisis recognition, predictive validation, casualty response, and refinement of primary and secondary damage simulation, the graphic visualization system will provide a first-order estimate of their accuracy.

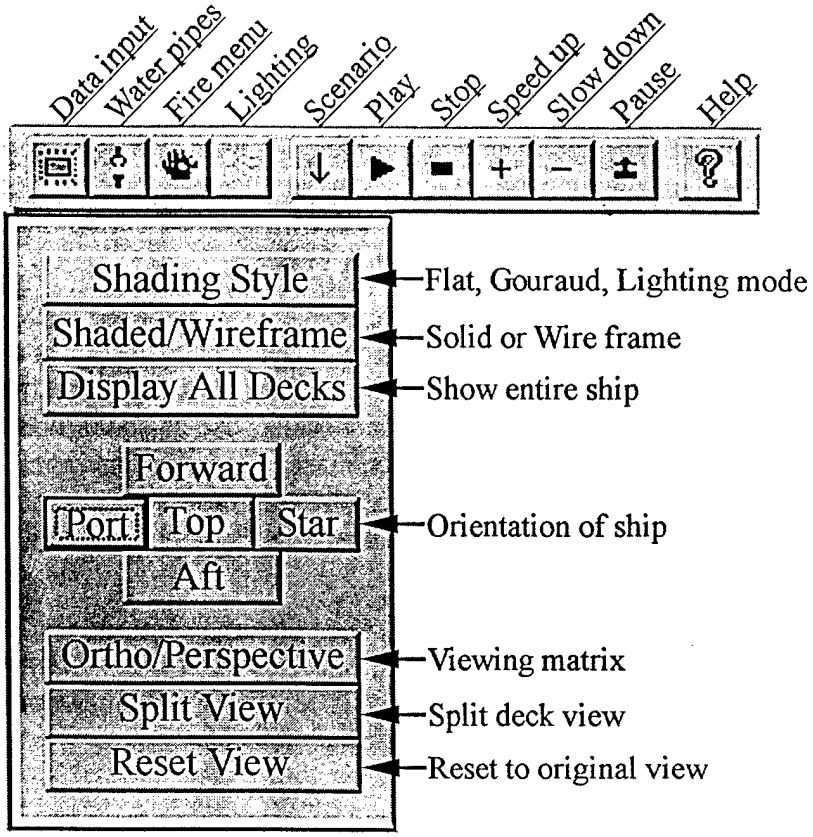


Figure 7. Illinois Ship Visualization System (figure 2 of 2): Enlarged view of the GUI visualization toolbar on the left side of the visualization screen

The visualization toolbar shown in Figure 7 allows the user to load a specific ship model, modify ship properties, start a crisis scenario or play back an earlier scenario and provides multiple viewing options for displaying the ship. The speed up and slow down option has the dual purpose of changing the speed of the simulation in progress, and quickly moving through the scenario in playback mode.

In the visualizations shown throughout this report, the status of each compartment is indicated by its color:

- Green/Gray = Intact;
- Yellow = Ignited;
- Red = Engulfed;
- Black = Destroyed.

Section 2.2.3 on primary and secondary damage shows examples of visualizations of crises in progress. Currently, smooth Gouraud shading is used to produce a color gradient between two compartments of different status. During the first year, this will soon change to uniform coloring, with different levels of shading to indicate the compartment's progress from one state to the next. For example, a compartment might be bright yellow when first ignited, slowly becoming more red as it approaches engulfment.

Further, color-coding could be added for any ship compartment states if this information would assist the damage control Supervisor/Operator with respect to Supervisory Control or making manual override decisions.

The visualization module is designed to go hand-in-hand with the Damage Control System (DCS) currently in use by following many of the visual conventions used by DCS, such as the use of standard Navy symbols. This should facilitate allowing an officer familiar with DCS to quickly learn to use the Illinois Graphic Visualization System.

The Illinois Graphic Visualization module interfaces exclusively with the SQL Database module. Even basic ship data, such as the coordinates of all compartments on a ship, is stored in the SQL database module, and not in the graphic visualization module. Any input that the user provides to the Supervisory Control system through the graphic visualization module, such as clicking a sector on the visualized ship, is input to the SQL database. The important point is that no domain specific ship data is hard-coded into the graphic visualization module. This maximizes the ease with which any graphic visualization system, such as the DCS, can be used with the Illinois Supervisory Control system.

The current graphic visualization system is based on the Open GL library, and hence can be run on a variety of platforms such as Windows NT PCs or Silicon Graphics workstations. On a Pentium Pro platform, we have found the best speed is obtained with the use of a PCI Matrox Millenium video accelerator board with 8 megabytes of WRAM.

3.2.2 Ship Specification Module

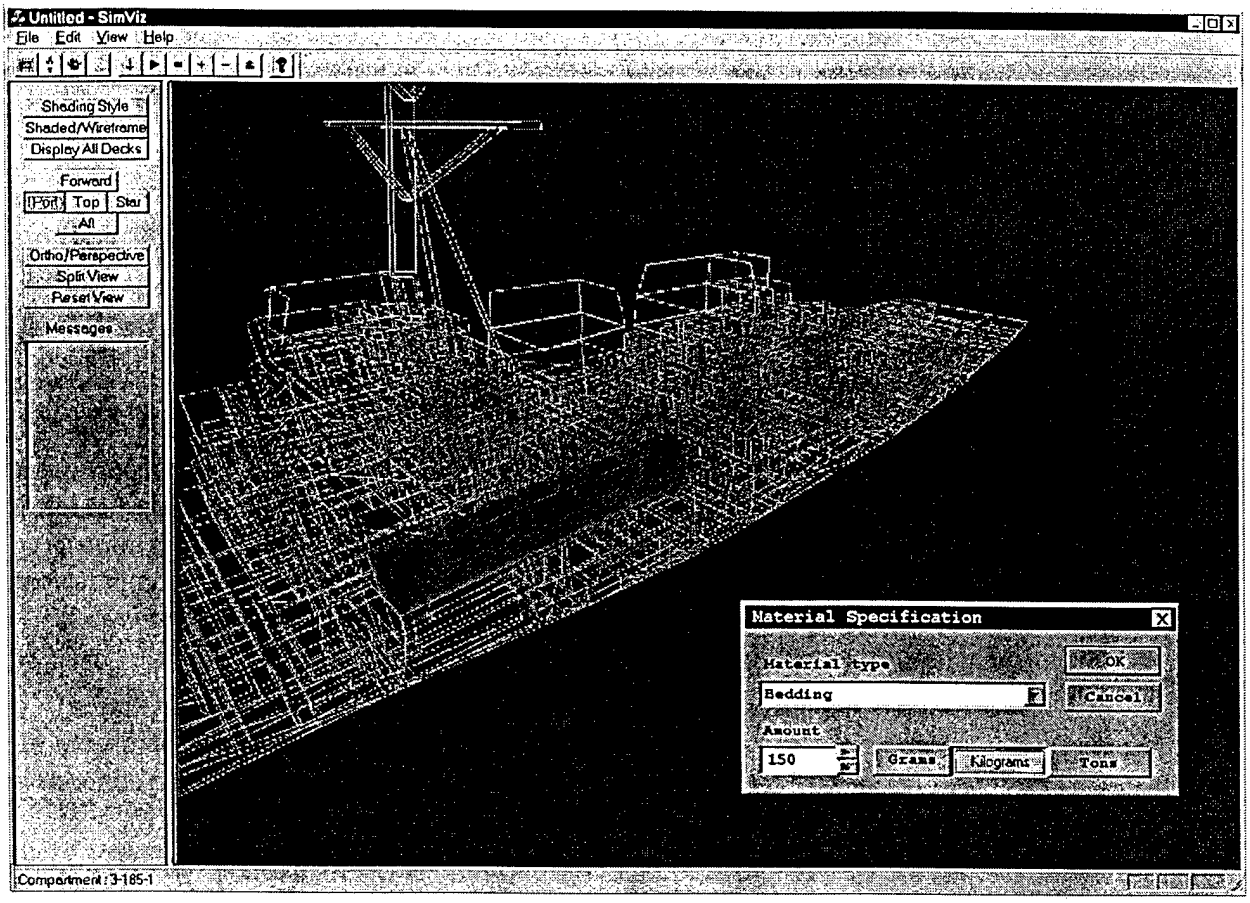


Figure 8. Illinois Ship Specification Interface (figure 1 of 2)

The ship specification module is a GUI interface (Figures 8 and 9) with which subject matter experts can specify and modify many aspects of the physical ship condition: materials in compartments, the topology of compartments, the insulation between compartments, the subsystems of the ship such as fire main and chilled water, the sensors and actuators that relate to damage control, and the ship deactivation information. For example, a combustibility expert can modify the amount of flammable materials in a compartment to see how this affects the fire spread. This ability is important both in the testing phase of the system in order to validate the accuracy of crisis simulation and prediction, and in practice aboard the ship when some change in its physical condition must be reflected in the computer model.

Ship specification is accomplished through a graphical interface that uses the same ship visualization model as all the other modules in this subsystem. The difference here is that instead of having a static model, it is possible to modify most aspects of the ship structure and its contents. For ship specification, there are two editing levels: ship-level and compartment-level. Ship variables are classified into these levels as follows:

Ship-level

- Fire main and chilled water systems
- Deactivation order of shipboard systems
- Compartment location and adjacency to other compartments

Compartment-level

- Insulation between adjacent compartments and wall thickness/armor level
- Types and amounts of materials in compartments
- Sensor and actuator locations
- Personnel locations

The process of ship specification as illustrated in Figure 8 is initiated by clicking on a particular compartment in the wireframe diagram (recall that an expanded view of the toolbar on the left is shown in Figure 7). Clicking brings up the Illinois ship specification interface, which allows viewing the contents of the compartment and making changes to any aspect of the compartment, such as amount of materials, thickness of the walls, amount of insulation between the walls, numbers of sensors in the room and their placement.

Figure 9 provides an alternative format for selecting a compartment to be modified. Once a compartment has been selected, the system will be able to display all aspects connected with it. And a graphical user interface will provide the means of making modifications to the contents of structure.

Of the above variables, the ship specification module currently supports material specification, as shown in Figure 8. The interfaces for modifying each variable will be highly graphical, so that changing a compartment-level variable would consist of selecting the compartment with the mouse and making the appropriate change in a dialog box and perhaps giving additional input with the mouse, such as selecting the location of a temperature sensor within the compartment or selecting a wall to specify the insulation type.

Deactivation information will be displayed as a deactivation diagram in a separate window. Editing the topology of the diagram will be accomplished by adding, deleting or moving the edges in the diagram to indicate a different deactivation order.

Altering the ship structure itself (making ship-level changes) will be accomplished via a CAD (computer-aided drafting) like interface, allowing the user to change the number of decks and add, delete and move walls, hatches, vents, pumps, valves and plugs, using the mouse and a "palette" of shipboard objects. This is important when structural changes made to the ex-USS *Shadwell* during the testing phase need to be reflected in the computer model of the ship.

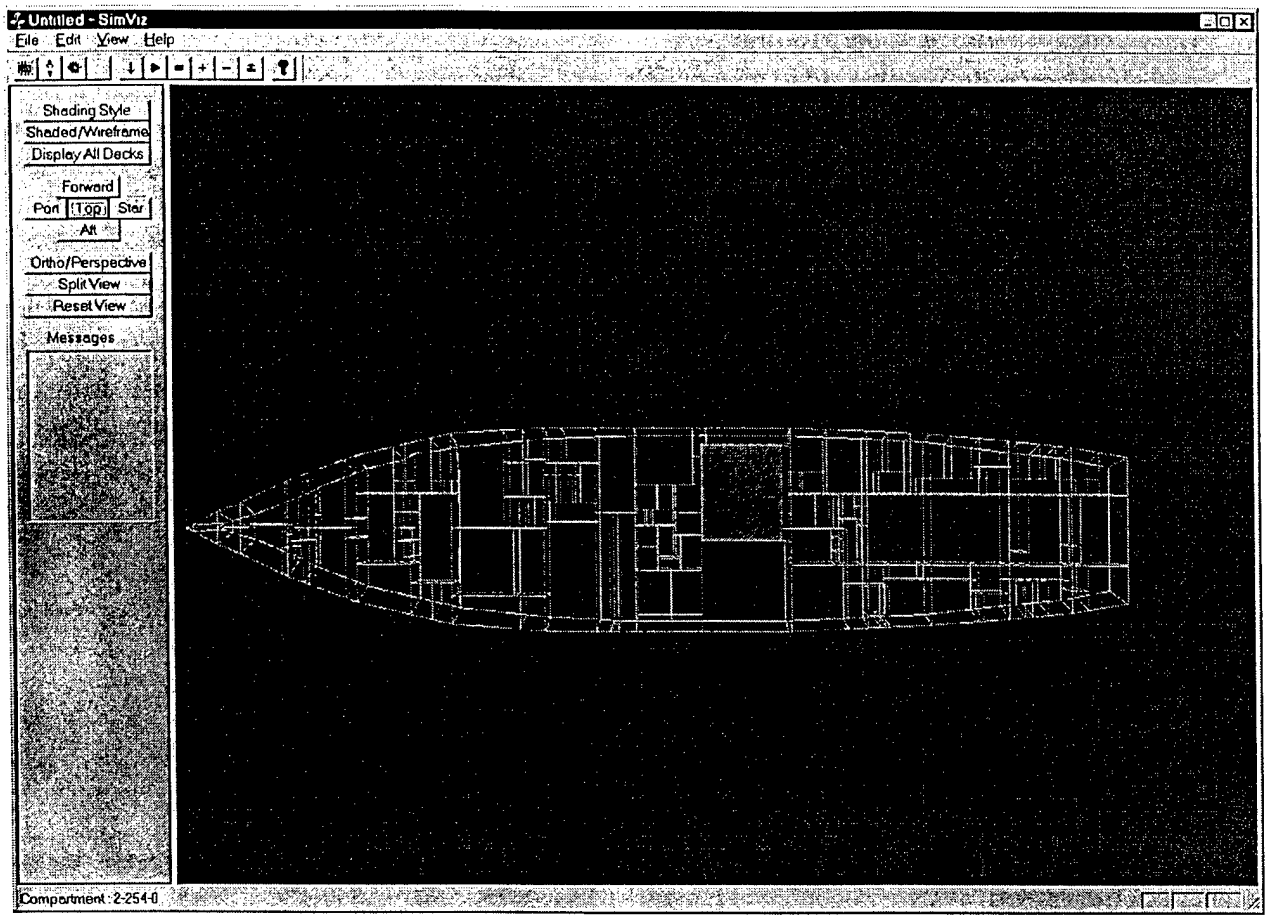


Figure 9. Illinois Ship Specification Interface (figure 2 of 2): Deck view. This provides an alternative viewpoint for observing and modifying the contents of compartments

3.2.3 Scenario Specification Module

The scenario specification module allows the user to put a ship in a specific crisis situation. It provides the ability to choreograph complex crises events with respect to their timing and nature. The two essential purposes of scenario specification are to test the entire system, as well as sub-modules throughout their development, and for the Crisis Predictive Validation Module to determine whether a suspected crisis meets the recommendations for a bona-fide crisis. Figures 10 through 13 demonstrate a sample scenario specification session. A crisis consists of one or more explosions or blasts/hull penetrations each of which can be assigned a particular compartment as its origin, as well as combustion parameters such as the force of the explosion, fragmentation level and impact angle. Some of this information can be entered visually using the 3-dimensional display. The blast information is then used by the secondary damage modules (Section 2.3) to determine the spread of the crisis over time.

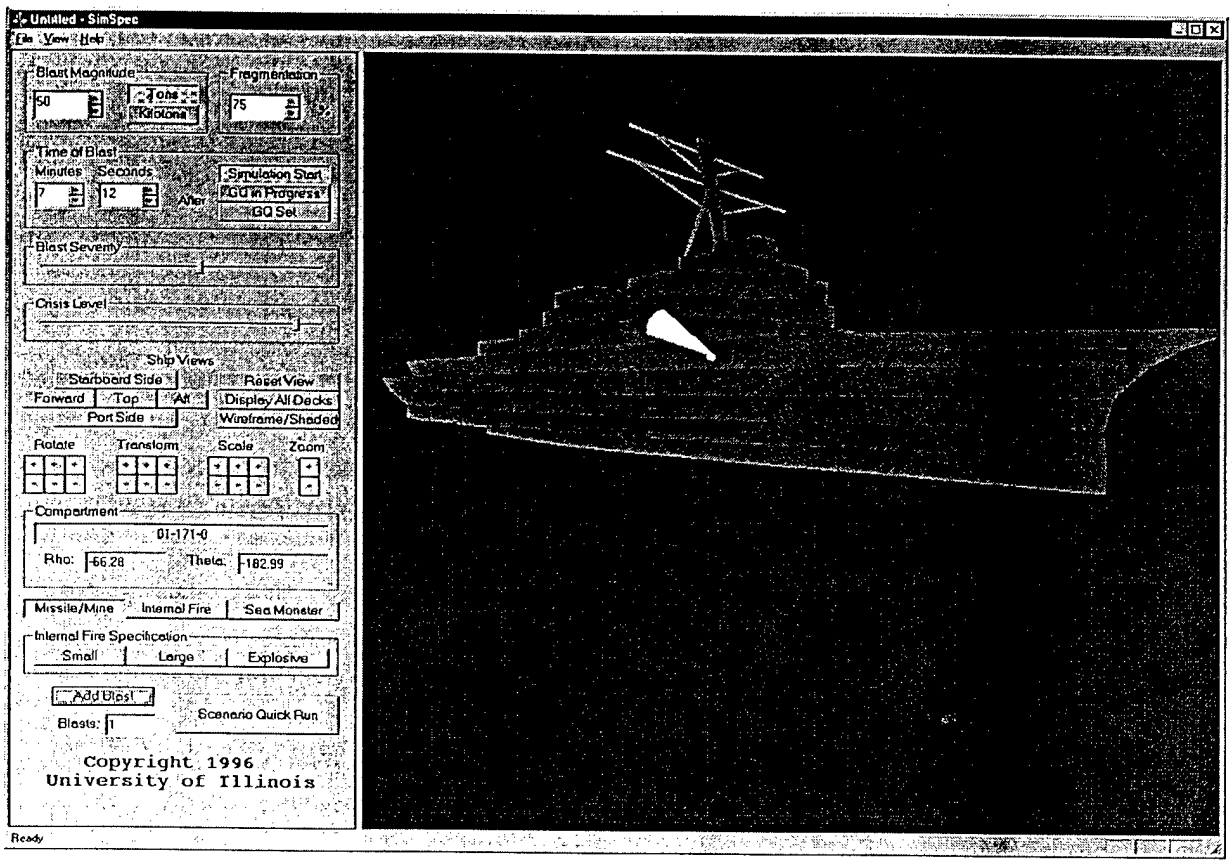


Figure 10. Illinois Scenario Specification Interface (figure 1 of 4): Selection of impact compartment by placement of a cone. See next figure for enlarged view of the toolbar on the left

Each scenario consists of one or more blasts; each assigned a specific time at which they occur. Once the user has specified a blast, he or she selects "Add Blast" on the toolbar (see Figure 11) to commit the blast to the scenario, and may then continue to add more blasts. Instead of specifying new blasts, the user will also be able to select blasts from previous scenarios and incorporate them into the current one. Once all blasts have been specified, the scenario can be saved to disk and then sent to the simulation module.

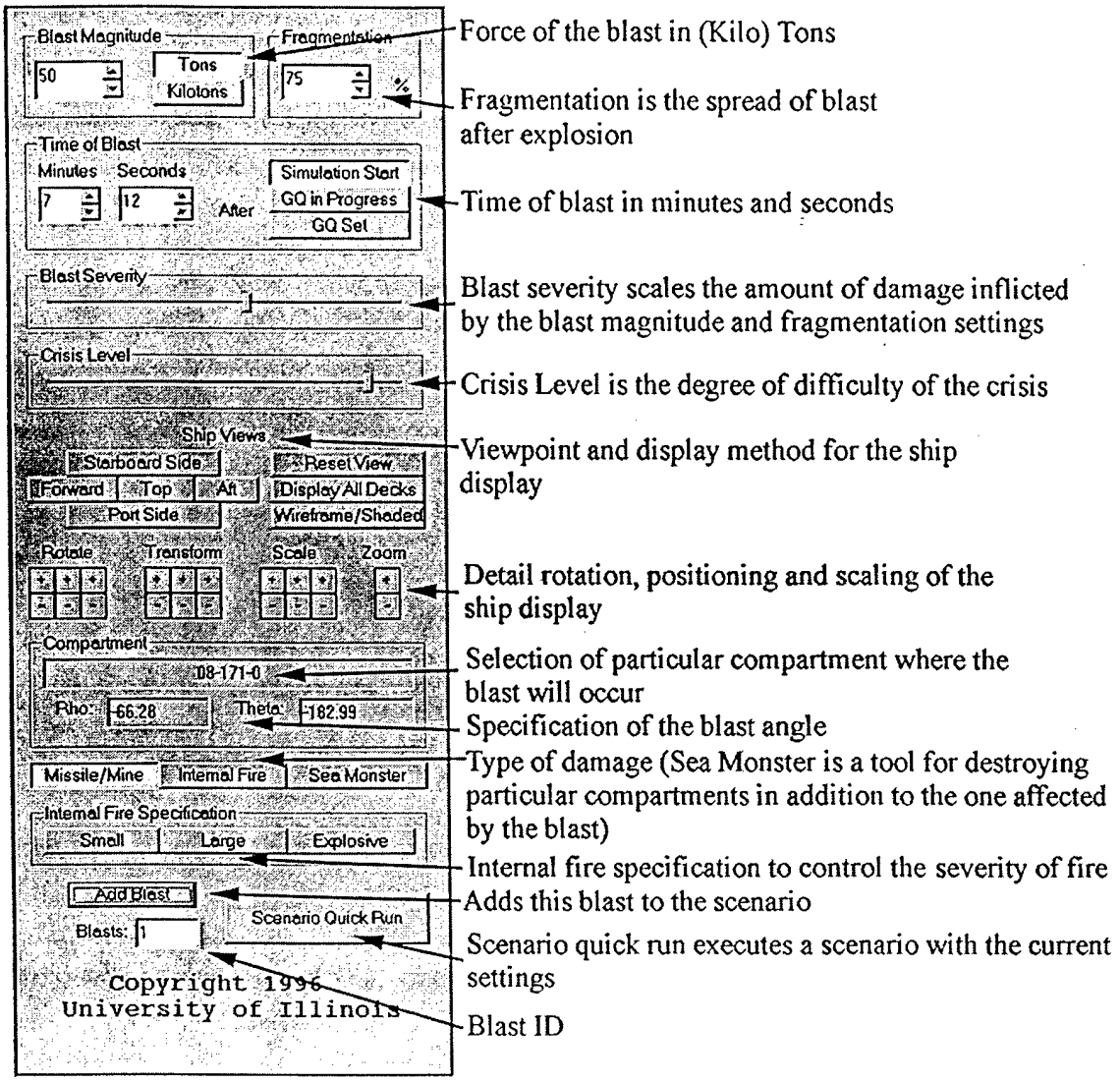


Figure 11. Illinois Scenario Specification Interface (figure 2 of 4): Detailed summary of specification toolbar

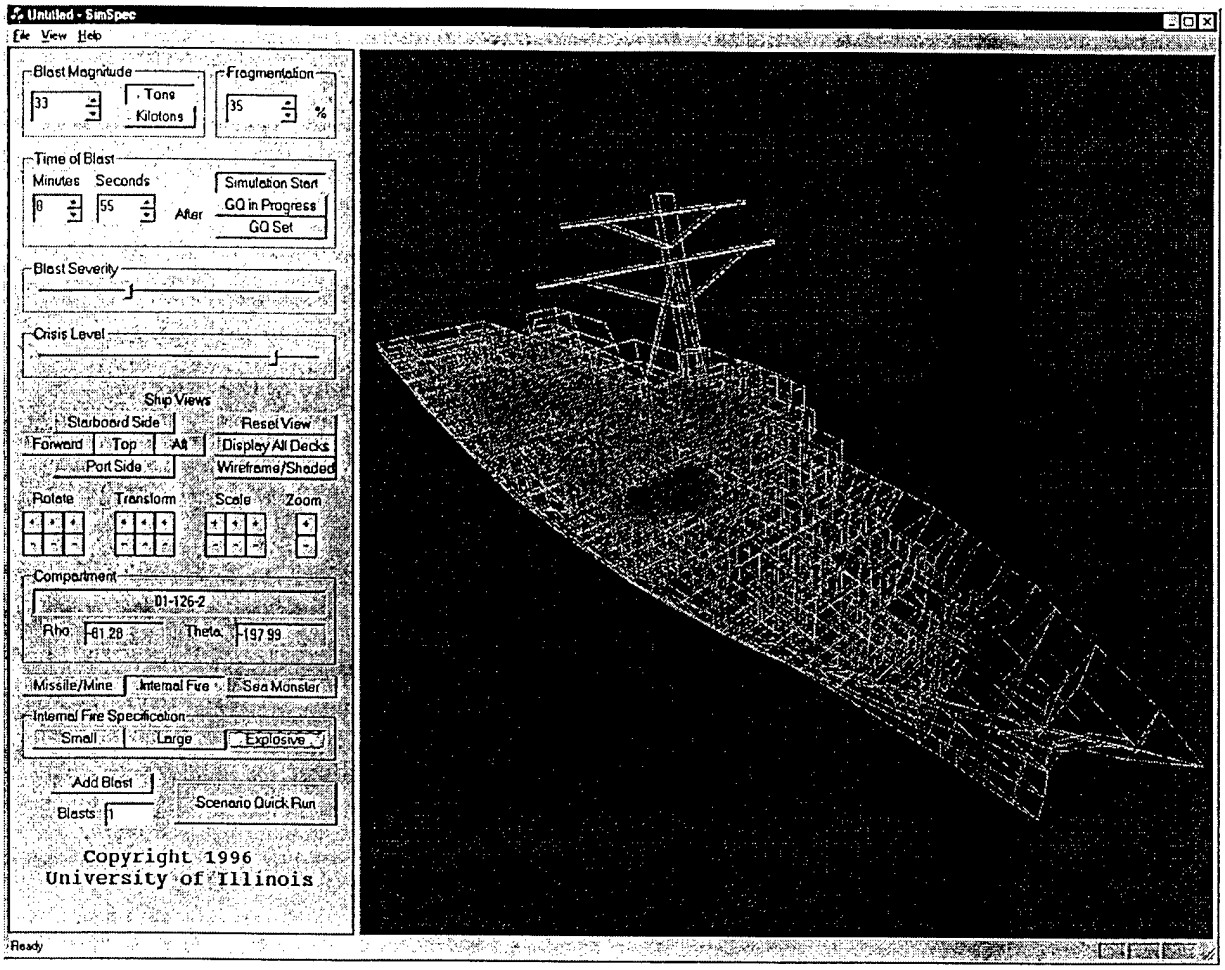


Figure 12. Illinois Scenario Specification Interface (figure 3 of 4): Wireframe view with selected compartment

The value of the wireframe representation of scenario specification (Figure 12) is the easy placing of initial blasts and other crisis events inside the ship rather than on the hull of the ship.

The fourth method of scenario specification is shown in Figure 13. The user can place an initial blast event with total precision. That is, a primary damage event that was calculated using the BDE system can be input to the Illinois crisis simulator by "painting" the damage vectors using the sea monster which allows altering the status of any room with respect to level of damage, amount of subsystem deactivation, flooding, fire level, and the like.

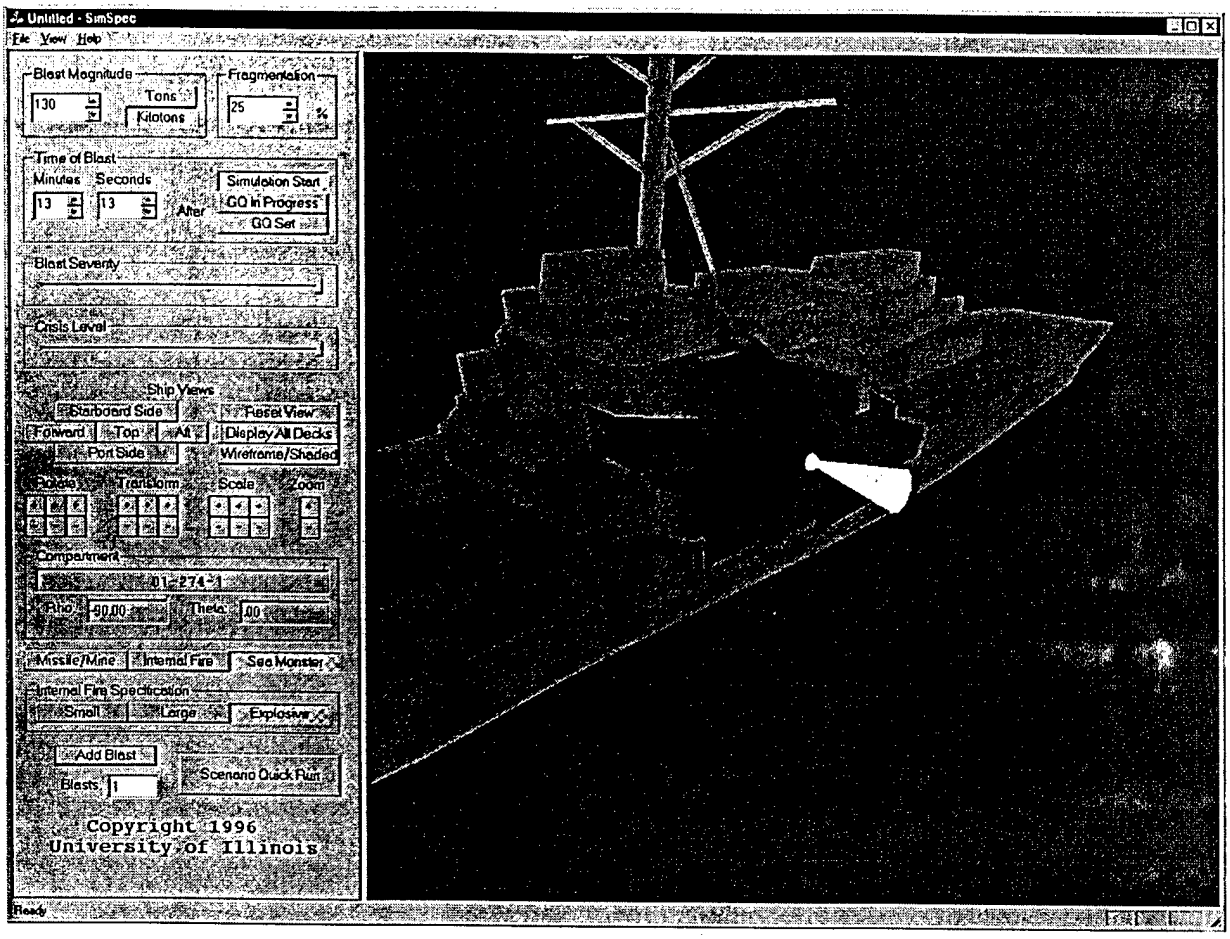


Figure13. Illinois Scenario Specification Interface (figure 4 of 4): Blast specifications with user specified damage shown in red. The "sea monster" was used to pre-specify damage to the blast region that will be cumulative with the impact vector's damage

The scenario specification module, especially the "sea monster" mode, plays a vital role in Predictive Crisis Validation. Recall that predictive validation begins when the Crisis Recognition System outputs a description of a suspected crisis. The Predictive Crisis Validation system is then charged with the task of validating the crisis. The first step in doing this is to use the Scenario Specification Module to set the ship in the crisis state determined by the Crisis Recognition System. The Simulator then carries the crisis simulation forward in time, and then the predicted simulated crisis values are matched against the new sensor readings.

3.2.4 Supervisory Interface Module

The supervisory interface module provides supervisory ship personnel with insight into the decision-making rationale of the automated system and provides the ability to override its actions. Most of our discussion of the Supervisory Interface is in Sections 3.6 and 3.7 of this report; these later sections describe the experimental laboratory tests that will assist in design modifications to ensure successful communication with the supervisor. This section focuses on the technical issues associated with the supervisory interface.

Figure 14 shows our first-pass implementation of the DC-ARM Supervisory Interface. This Interface was implemented using MacroMedia Director 5.0. The primary principle of our design is that all necessary knowledge must be transparent and visible in the interface. A secondary principle is that the supervisor has options regarding the type and mode of display of information. There are three panels:

- A display of sensor and actuator values in three modes: crisis level, ship view, and sensor type
- A display of Supervisory Control conclusions: crisis recognition, predictive validation, and casualty control.
- A panel for manual override that provides information necessary to modify any of the above sensor data or Supervisory Control conclusions

These three components of the Damage Control Supervisor Console are illustrated in Figure 14. The first display—for sensor data—appears at the top. The supervisor can control the types of sensors displayed at any instant, with options for display by crisis mode (all, abnormal, or critical), ship view (whole, deck, or compartment), and sensor type (temperature, flooding, rupture, hull, or passages). For instance, by selecting "CRITICAL," "WHOLE," and "FLOODING," all critical sensor messages related to flooding for the whole ship will be displayed. By selectively choosing the viewing criteria, the supervisor has instant access to any information on the ship, and thus can pinpoint a problem. Sensor messages are displayed only if they are extreme (e.g., the temperature exceeds a critical threshold). The supervisor can select one of three modes for display of the desired sensor data, allowing it to be viewed in a text message panel, with the DCS system or with the Illinois Graphic Visualization system.

The simulator uses the sensor data to form conclusions about events occurring in the ship. These conclusions are displayed in the second or middle panel. Conclusions about crisis recognition and predictive validation are presented via text in windows. A third window presents the system's preferred and alternate casualty responses. Any of these conclusions can be overridden by the supervisor through use of the manual override unit on the console.

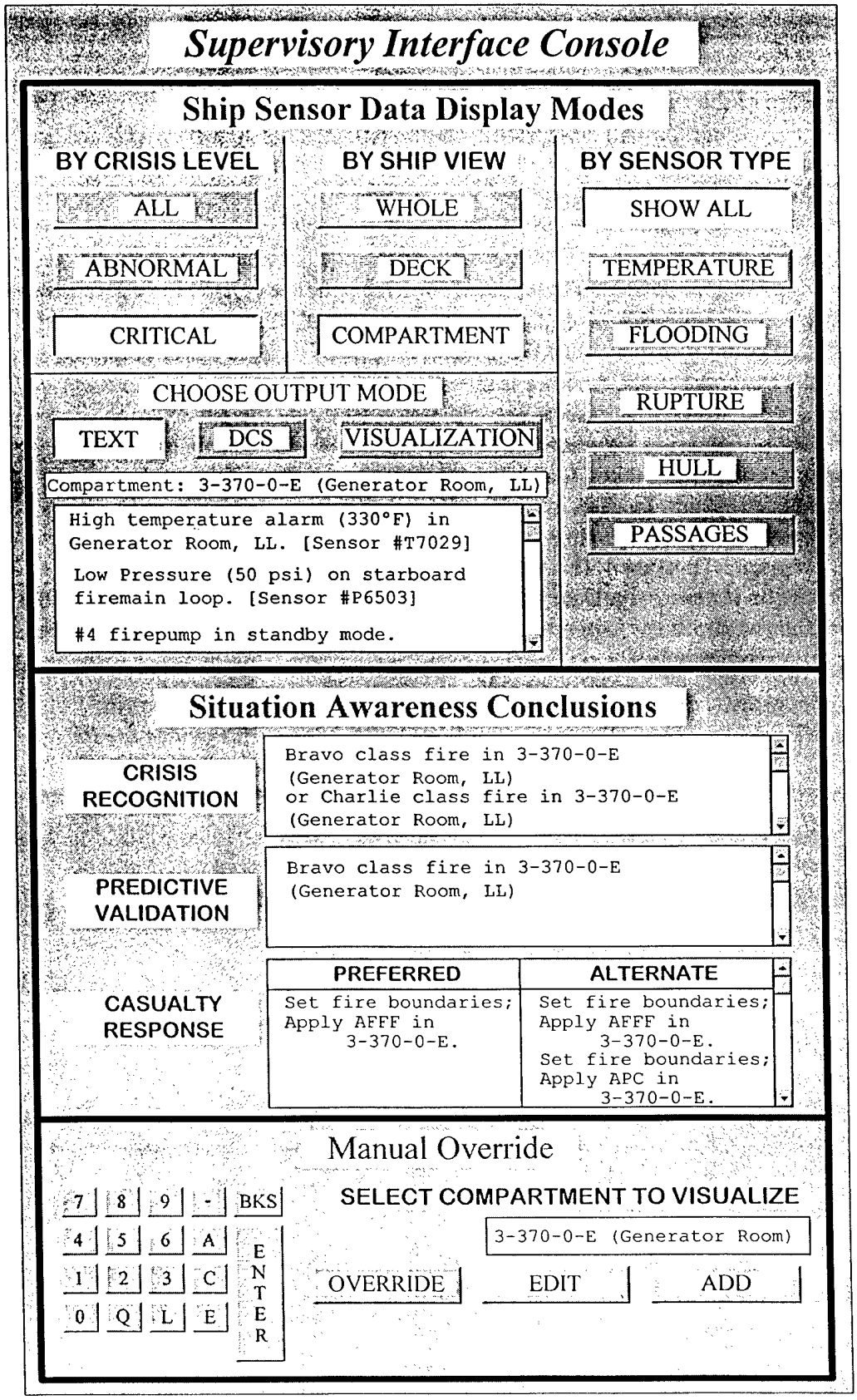


Figure 14. Damage Control Supervisor Interface

An important feature of the supervisory control interface is that everything that is needed to view and control the simulation will be available on one screen. This control is affected by means of the manual override panel. For both displays of sensor data and Supervisory Control conclusions, the supervisor has the option of editing messages and adding new data. With sensors, there will be additional option to change their state (e.g., turn on or off). If the supervisor wishes to carry out a supervisory control decision (see Section 2.6.3), it can be implemented by choosing an option on the manual override panel (e.g., EDIT), and following instructions. There are many associated displays that will be available to the expert, which work in conjunction with the Supervisory GUI Interface. One of these is a 3-D graphical visualization of the evolving crisis. Another is Damage Control System (DCS) module, which was developed by CAE Electronics. These displays facilitate the presentation of a global view of an evolving crisis situation. The crisis supervisor will be able to view exactly what actions are currently being taken (or recommended) to manage the crisis, as well as what actions were taken at any point in the past. The supervisor will have the ability to change or cancel any actions the system might take, as well as order additional actions he or she feels are appropriate. This means that ultimately the crisis supervisor is still in full control of the situation if necessary. Data with human subjects will be collected to refine the multimedia interface as needed to support control decisions by the supervisor. (See section 2.6 for further discussion)

The supervisory console shown in Figure15 represents the present state-of-the-art in providing Supervisory Control of Damage Control in a DCA setting. It provides the DCA with the information and commands relevant to carrying the task of *coordinating* ship damage control. This console was created at Illinois using Macromedia Director and is able to run over the Web on an Intranet. It is adapted from the Integrated Damage Control Team Trainer (IDCTT) system developed earlier by Tekamah and reimplemented at Illinois.

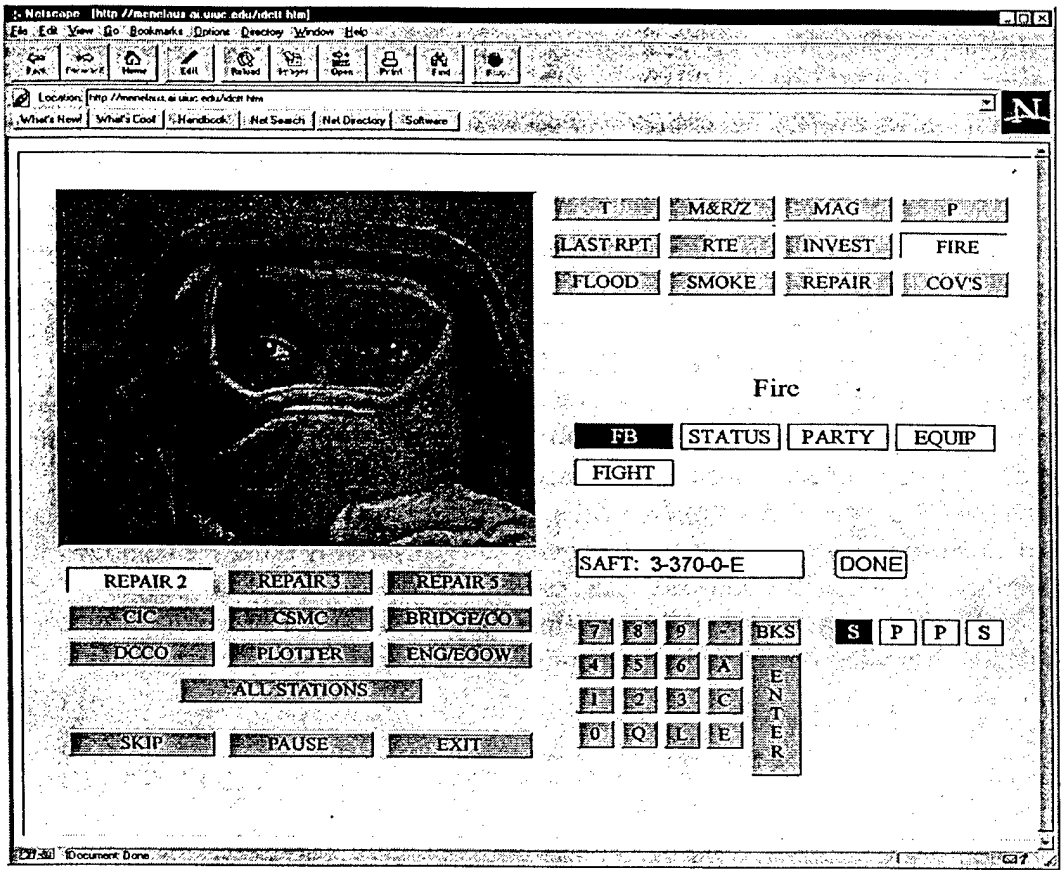


Figure 15. Example of a Damage Control Assistant Supervisory Control Console, for coordination of repair teams, that works in conjunction with DCS

Our evaluation of the appropriateness of the IDCTT DCA interface for our needs led us to modify it in a number of ways. First, we needed to generalize it. The Tekamah version was specialized to allow solving a single scenario while the Illinois project utilizes a general-purpose scenario generator. With the modified interface, the DCA has the option of issuing commands to the computer simply by speaking to it. A speech recognition engine processes all speech and then gives these commands to the computer. Depending on the outcome of the action, the computer can respond verbally, using text-to-speech synthesis. In addition, the computer can inform the expert of different situations when they occur using text-to-speech. Our version of the DCA Console interface was implemented in Director 5.0 and can be accessed via a web interface. Figure 16 shows this console. Our experience with this damage control interface can be expected to be an asset in the design, testing, and modification of the supervisory interface for DC-ARM.

3.2.5 Web Interface Module

The Web interface module provides the means to run the Supervisory Control system over the Web. First, it allows easy access either on or off ship for damage control simulation. Secondly, it gives materials researchers the opportunity to investigate combustibility from practically anywhere, permitting them to update their codes and materials models.

The module is comprised of all the GUI and visualization interfaces (e.g., ship specification, scenario specification, 3D graphics visualization, Damage Control System (DCS) and supervisory control). All of these features will be readily available simply through the use of World Wide Web browsers such as Netscape and Internet Explorer. Both the ship and scenario specifications can be input via a form on a Web page. All 3-D visualization would be implemented through a plug-in to the WWW browser. In addition, the supervisory control, implemented using Macromedia Director and placed on the Web with Shockwave, will interface with the DCS through a set of C++ calls. To control the interface, the user will have the option of using speech recognition, thereby being able to verbally order the computer to perform specific tasks. Then, the computer can respond vocally using text-to-speech synthesis. The 3D ship visualization, DCS and supervisory control will be continually updated as new information dealing with a scenario becomes available.

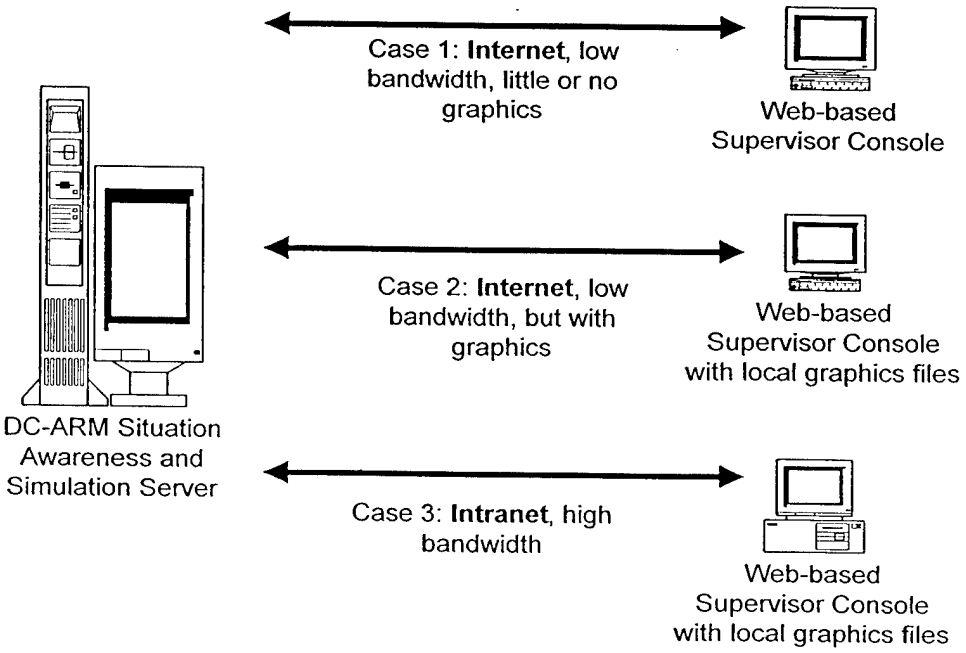


Figure 16. Either over the Internet or an Intranet, Web-based clients connect to the server, which is responsible for all the calculations and running the simulation

The Web interface module is geared toward two network types, specifically an Intranet and the Internet. With an Intranet, a specific site will have a local server that processes all the simulations, sending information back and forth with the client, a machine that has a Web browser and any plug-ins and additional files necessary. An Intranet allows for fast access to the server, and transferring large files in a matter of seconds. For the Internet platform, there

are two cases. All clients over the Internet need to have a WWW browser and any additional files, but those with slow connections, will have the option of downloading additional files to increase the speed of the connection to the server. With the latter option, however, some functionality may be lost. It is important to note that in all platforms the client has to do virtually no work besides watching and participating in the simulation. The server does all the computation and holds the knowledge base. This type of design allows a client to run a simulation from virtually anywhere.

The Web interface module gives the crisis supervisor options that are not currently available. Not only can the system be used on a ship by the crisis supervisor, but also an observer could watch a simulation or real-life crisis from a remote location and aid in crisis management.

3.2.6 Ship Interface Module: Sensor and Actuators

This module is the mediator between the DC-ARM system and the physical systems on the ship. Input to the Supervisory Control system comes via the ship's sensors, which can be organized into the following categories:

- Temperature
- Flooding
- Holing (i.e., hull or bulkhead breaks)
- Hatch and door closures
- Fire main pressure and flow
- Ventilation ducting

The intended implementation of these sensors is described in CARDIVNSWC-TR-85-95/01 [Whitesel & Nemerich, 1995]. Output is in the form of messages to various ship stations and commands to physical actuators. These actuators might include:

- Hatches and doors
- Sprinkler systems
- Halon
- Valves and pumps

and any other relevant systems that can be remotely controlled. The exact method by which the sensors and actuators would be connected to the Supervisory Control system is still under revision, although it is likely that this procedure would follow standard Navy practices for shipboard electronic communications.

3.3 Subsystem 2: Physical Ship Simulation Subsystem

The physical ship simulation is responsible for numerical simulation of physical events that relate to a crisis. It models the spread of fire and smoke, flooding and fire main rupture, as well as stability and buoyancy. It calculates, for example, the pressure at any point in the fire main or cooling system. It calculates the time to engulfment of a compartment by fire. And by its simulation of flooding and rupture events, it is able to provide input to the Navy's stability algorithms. A unique feature of the ship simulator is that human or automated actuators influence the course of events. The spread of fire is affected, for example, by the flooding of compartments.

The physical ship simulator plays three key roles in the automated Supervisory Control system. First, it generates training examples of crises that can be used by neural and symbolic learning algorithms to refine the knowledge base of the crisis recognition system and the casualty response system. Second, the physical ship simulator is the key element for predictive validation. The ship simulator is initialized with the parameters of a suspected crisis, and the crisis is confirmed if the crisis grows over a short period of time at the rate predicted by the physical ship simulator. One of the four PCs of the Supervisory Control system is devoted to Ship Simulation.

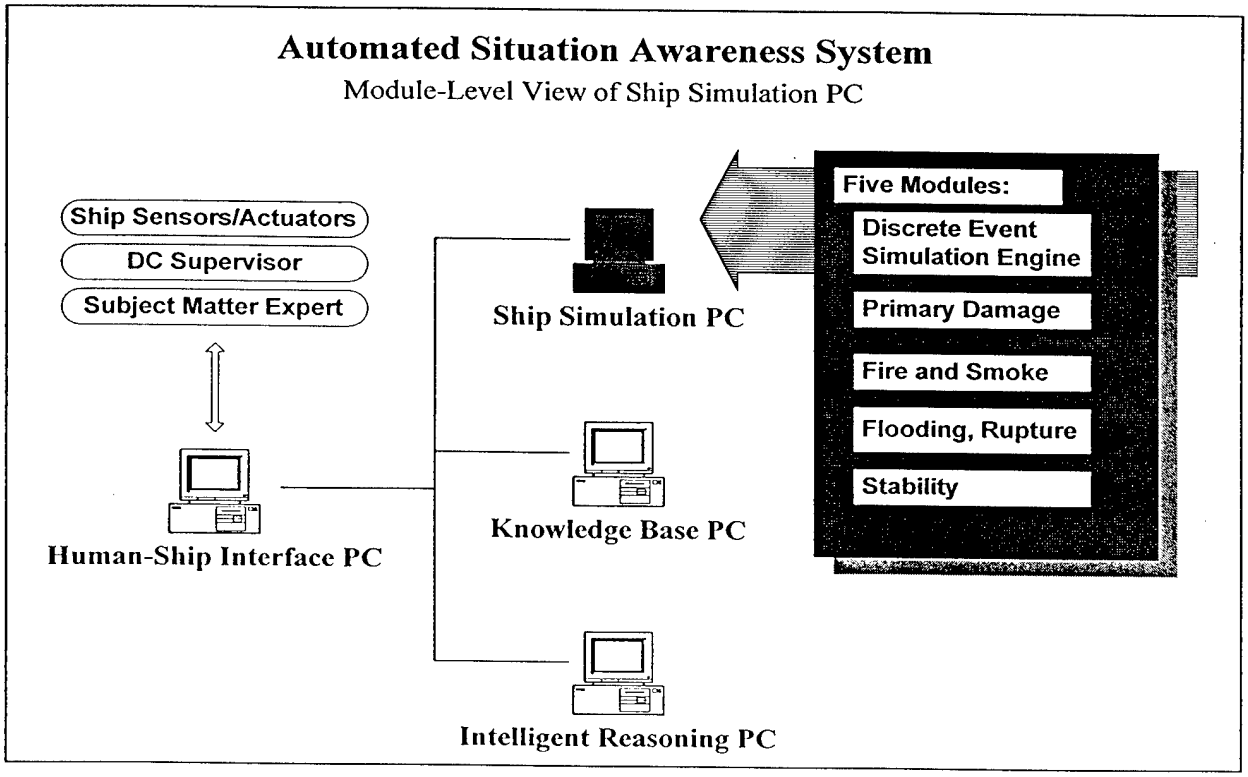


Figure 17. The Ship Simulation Subsystem and its five major modules

3.3.1 Discrete Event Simulation Engine

The discrete event simulation module provides a generalized framework for modeling the behavior of physical processes, such as fire spread, and user interaction (flooding compartments, closing hatches) on the direct physics of the ship.

The simulation engine maintains and updates the state of the system (ship) by responding to interactions of external and internal agents. These interactions are modeled with time-stamped events. The simulation engine schedules the events by maintaining event queues. In addition, the module allows the actions of events to modify the behavior of other events, such as extinguishing a fire that has ignited a compartment. The interface to the simulation engine is designed to support interaction with independent modules.

The interrelationships between the modules of the ship simulator, and modules external to the ship simulator, are illustrated in Figure 18.

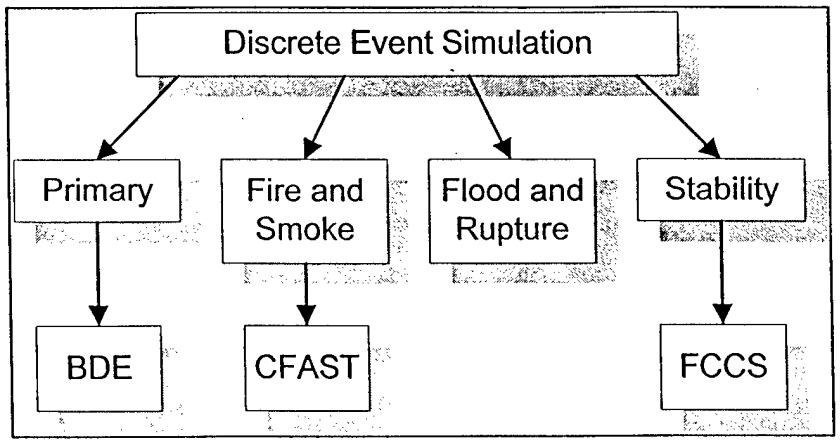


Figure 18. Discrete Event Simulation

The discrete event simulation provides a layer of abstraction between mathematical models and changes in the state of the system. The module will take advantage of concurrency through parallel and distributed methods. In order to maintain the accuracy of the module, synchronization techniques within the system are necessary. The module explores both conservative [Chan, 1979] and optimistic strategies of synchronizing processes. In particular, the module will implement an optimistic strategy with error detection, using the rollback mechanism introduced by Time Warp [Jefferson and Sowizrral, 1985]. The module will also examine failure detection and recovery methods in asynchronous systems as well as state saving mechanisms such as copy state saving [Bauer and Sporer, 1993]. The module will compare synchronization methods through a transparent monitoring process.

The discrete event simulator module coordinates the primary and secondary damage simulation modules. Figure 19 shows a compartment state transition diagram for secondary damage.

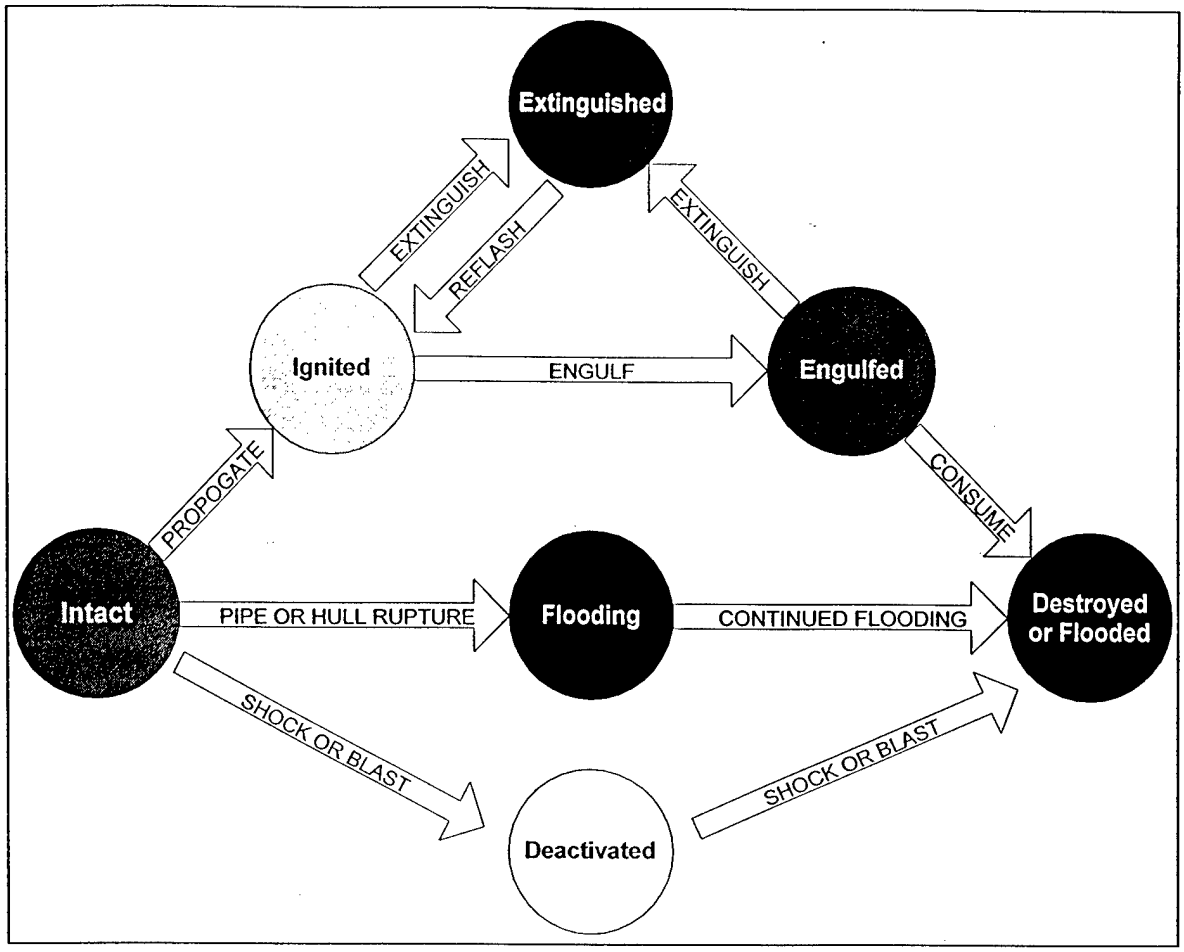


Figure 19. Major compartment states and their transition

The initial state of all compartments is *intact*. When fire propagates to a compartment its state changes to *ignited*. If the fire is allowed to burn, the compartment can move into an *engulfed* state, where it is capable of propagating fire to its neighbors, and finally, once all combustible materials have been incinerated, become *destroyed*. If fire-fighting efforts are undertaken, or if the oxygen is significantly reduced, the fire can become extinguished. A re-flash, however, can cause the compartment to become ignited again.

The simulation of a crisis is broken down into two major sub-modules: the Primary Damage sub-module, and Secondary Damage sub-module. The Primary Damage sub-module models the initial damage caused by the crisis. The Secondary Damage sub-module builds upon this information to simulate the progression and expansion of the crisis.

3.3.2 Primary Damage Module

The Primary Damage Module essentially deals with the initial damage caused by an explosion resulting from a mine or missile hit. The initial impact of the explosion destroys, engulfs, and ignites compartments on the ship. In order to model this part of the crisis, the primary damage algorithm splits the force of the explosion into x, y, and fragmentation components (see Figure 20). The fragmentation component is calculated from the fragmentation level as determined from the weapon specification. Damage is calculated by breaking the force on a given compartment into its components and dampening the force in

the specific directions. According to the algorithm, all compartments where at least one component of the initial impact force is not zero are considered destroyed. Furthermore, the boundary compartments (those on the fringe of the explosion) are deemed to be engulfed by fire, and their neighbors are ignited. Figure 21 shows the Visualization Module's representation of primary damage.

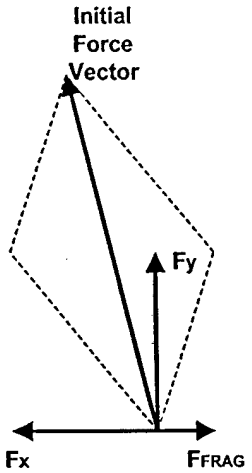


Figure 20. Calculation of initial impact force components

3.3.3 Secondary Damage: Fire and Smoke Module

Fire on a ship at sea is an extremely dangerous phenomenon. The smoke and toxic gases, as well as the intense heat, threaten the lives of the crew. In addition, the spread of fire to strategic spaces, such as the magazines, engine spaces and control rooms, can result in explosions that cause crippling structural damage, loss of propulsion, and/or command and control. For these reasons the ability to predict the spread of fire and smoke is vitally important to successful damage control. Our secondary damage module for fire and smoke simulates the spread of fire and smoke throughout the ship. CFAST, a zone model program, [Peacock et al., 1993] is used to model combustion in each individual compartment. Given the dimensions and ambient conditions of a burning compartment as well as its contents, CFAST produces data on the progression of the combustion process, propagation times to neighboring compartments and generation of smoke.

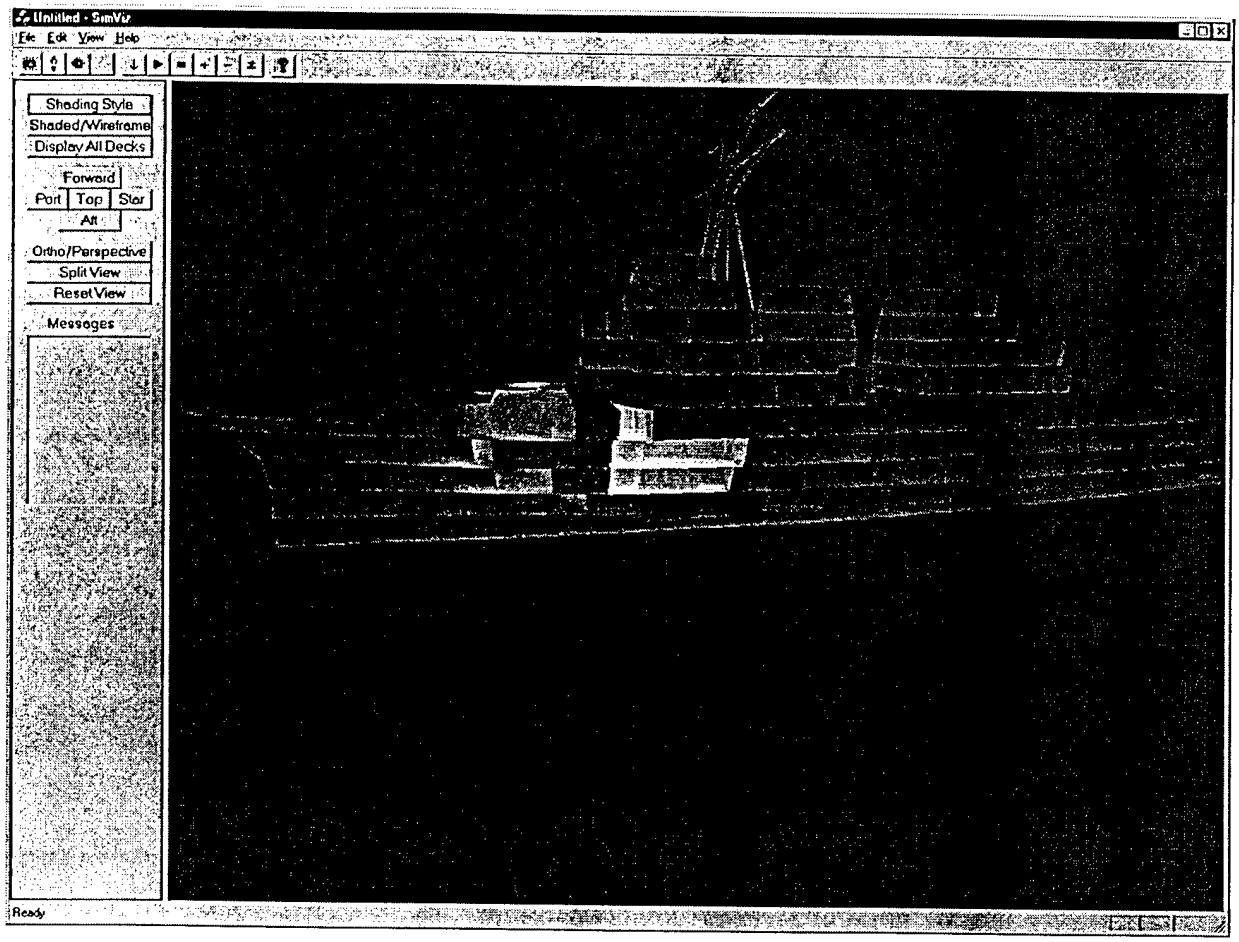


Figure 21. Primary damage with secondary damage in progress

In Figure 21, white compartments indicate immediate destruction during primary damage. Red and yellow indicate engulfed and ignited compartments, respectively, as a result of secondary damage. Black compartments, in this view, are intact and undamaged.

With respect to fire, our model recognizes five possible states for a compartment: *intact*, *ignited*, *engulfed*, *destroyed*, and *extinguished*. Transition between these states is modeled by the Discrete Event Simulator, discussed further in Section 2.3.1. CFAST is executed for a given compartment when it becomes ignited. The output specifies the times (in simulation units) to engulfment and destruction, respectively, as well as three propagation times to neighboring spaces: through the bulkheads, overhead and deck. Figures 22 and 23 show fire spread through the ship.

Our fire and smoke simulation module also models the effects of both passive fire boundaries (such as bulkhead insulation) and direct fire-fighting activity (such as sprinkler systems) by the dynamic scaling of time-to-compartment-status-change values output by CFAST. These time values can be recalculated at any point in the combustion process -- whenever a retardant (such as water or AFFF) is introduced into the compartment.

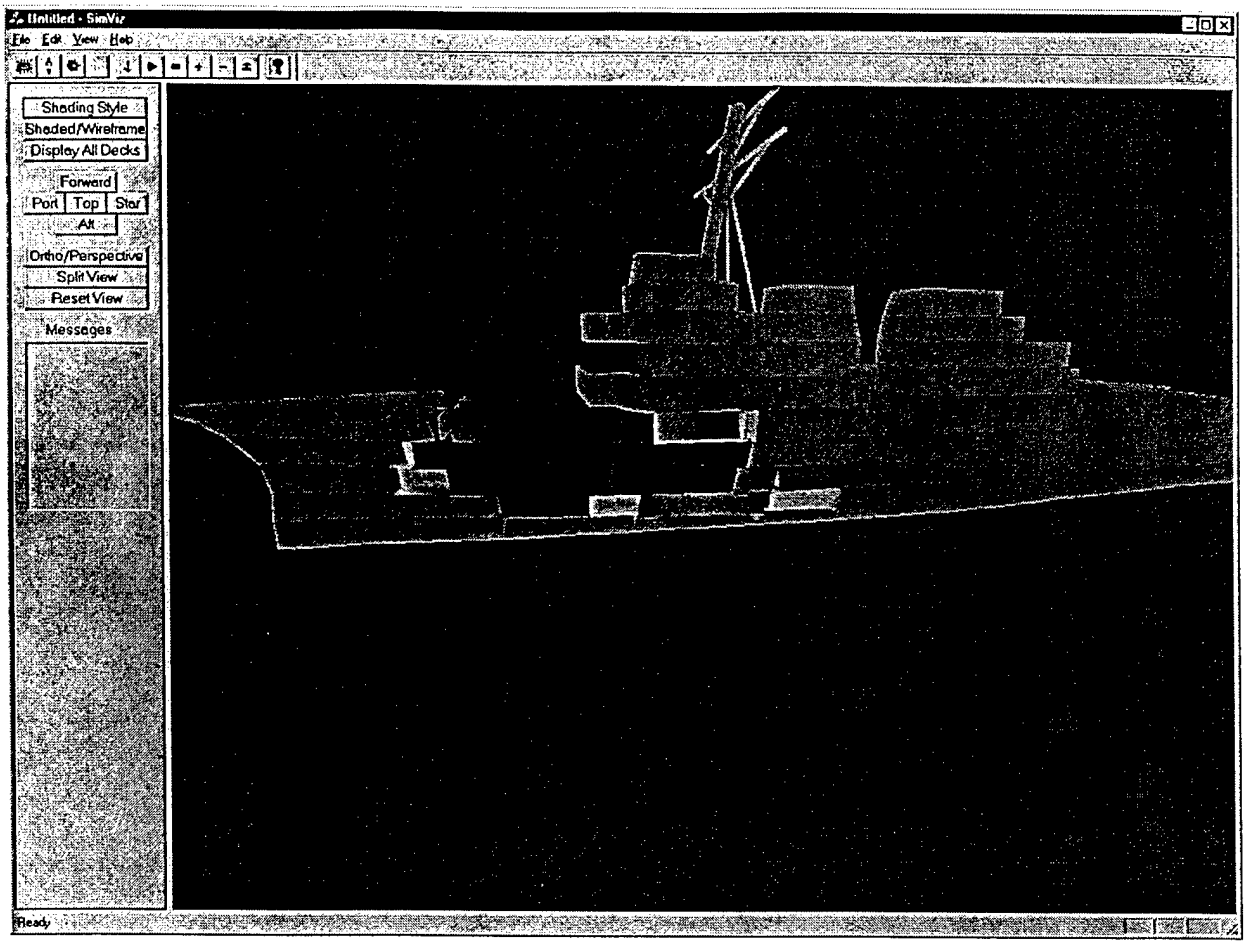


Figure 22. Secondary damage

In Figure 22, black compartments are destroyed, red are engulfed, yellow are ignited and green are intact.

Due to the networking advantages offered by the Microsoft Windows NT distributed operating system, our overall system is designed to run under this environment. However, the standard PC version of CFAST only operates under DOS. It is possible to invoke a DOS program from Windows through a DOS shell, but this approach is highly inefficient and not practical for fast, real-time scenario generation. Instead, we have invested a great deal of effort into porting the original 16-bit code to 32-bit Windows code. Furthermore, we have incorporated the CFAST program into our system as a static library, instead of a separate executable. These changes provide us with a substantial speed-up in execution, as well as more compact code, and permit the real-time modeling of fire-spread for each ignited compartment.

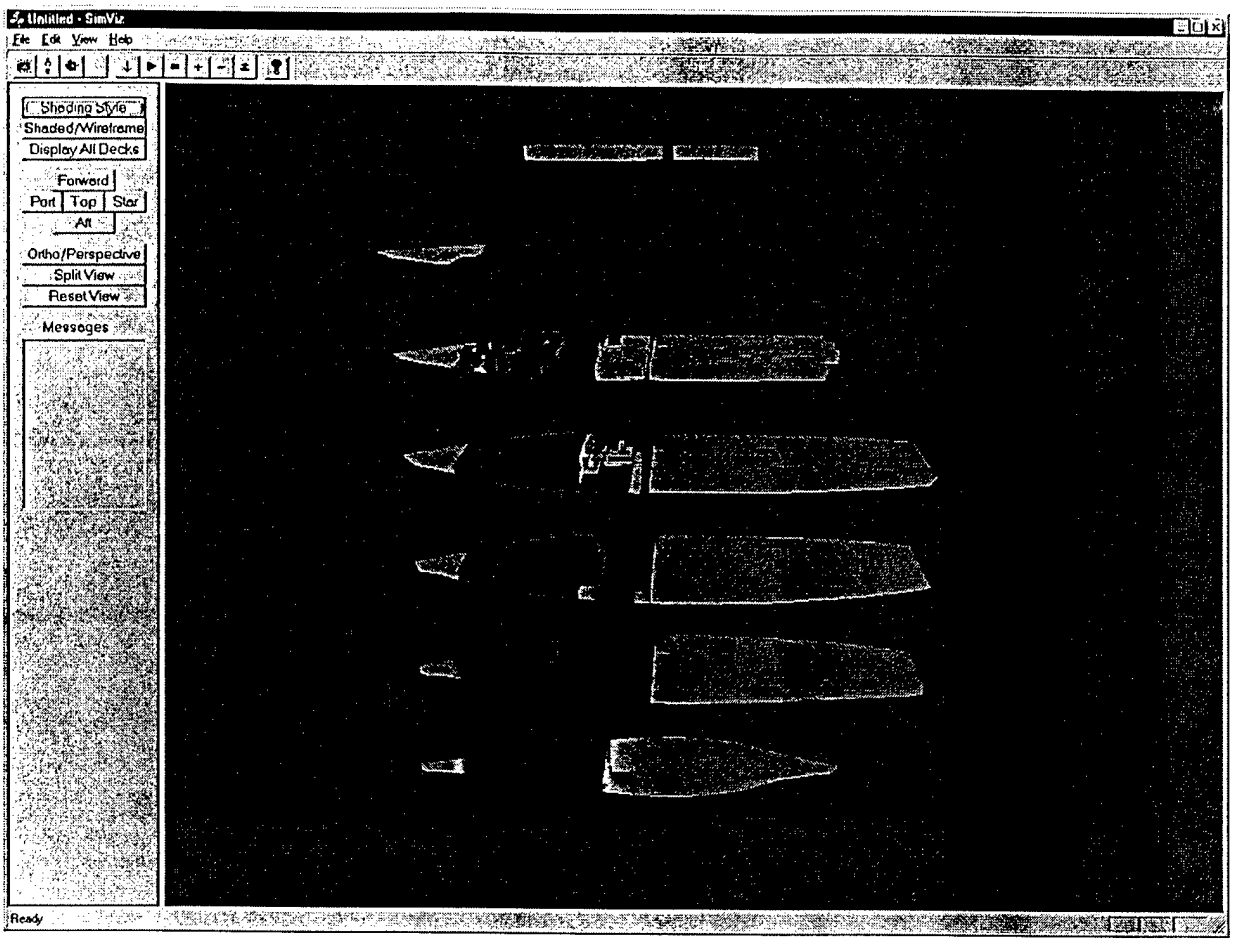


Figure 23. Secondary damage split view

We also model the simulation of smoke propagation through the ship. The generation of smoke in all burning compartments is modeled by CFAST. The simulated spread of this smoke to other ship spaces is accomplished by a network flow algorithm. To this end, all affected compartments are represented by a dynamically changing, undirected graph. All nodes representing burning compartments are deemed smoke flow source nodes and are marked as such. Additionally, all those compartments on the fringe of the smoke-engulfed area are considered smoke flow sink nodes and are also marked. The algorithm propagates smoke from each source to all reachable nodes in the graph and from all nodes to each reachable sink. This results in at least one inflow rate value and at least one outflow rate value for all nodes, except sources and sinks. Then, by the principle of superposition, these values are summed to produce a net flow rate for that node (compartment). Multiplying the flow rate by the simulation time unit gives the net amount of smoke that has entered or left the compartment during that time. From this, the current smoke concentration for the given compartment can be easily found.

The physical models of smoke propagation are based on those in the CFAST manual. There are three distinct cases:

(1) ***Horizontal flow through vertical vents.*** The general form for the velocity of the mass flow is given by

$$v = C\sqrt{\frac{2\Delta P}{d}},$$

where v is the mass flow velocity, C is the constriction coefficient and is approximately 0.7, d is the smoke density on the source side, and ΔP is the pressure change across the interface.

(2) ***Vertical flow through horizontal vents.*** This situation is different from (1) in that the temperatures are different across the interface, which causes the flow to be unstable. The overall flow rate is given by

$$\frac{\Delta m}{\Delta t} = C\, f(r,\varepsilon)\sqrt{\frac{\Delta P}{d}}\, S,$$

where C=0.68+0.17•ε, $\varepsilon = \frac{\Delta P}{P}$, f is a weak function of both r and ε and S is the area of the vent.

(3) ***Forced flow*** (e.g., fans blowing air through ducts). The flow rate is given by

$$\frac{\Delta m}{\Delta t} = G\sqrt{\Delta P},$$

where $G = \sqrt{\frac{2d}{C}}\, S$, C is a correlation constant dependent on the types of duct and fan, and d and S have the same meaning as those in (1) and (2).

3.3.4 Secondary Damage: Flooding and Rupture Module

The secondary damage flooding and rupture module keeps track of water levels and water propagation in flooded compartments, as well as the water pressure in all parts of the fire main systems. In the case of flooding, the main potential sources of water are hull ruptures below water line, pipe leaks and ruptures, as well as direct fire fighting. Clearly, hull ruptures are the most severe sources of flooding, but pipe ruptures and fire-fighting efforts can also introduce substantial volumes of water into ship spaces. Flooding can have severe effects on ship operations, such as restricting movement through some ship spaces, shorting electrical systems, and even affecting the buoyancy and stability of the ship.

The ship's fire main, as the name implies, is the main water-carrying artery on the ship. The fire main rupture model is important for two reasons. In the first place, it continually calculates the water pressure in every pipe segment of the system. This permits real-time simulation of pressure effects on the fire main of such events as pipe ruptures, sprinkler system activation, fire pumps going on and off line, as well as attaching fire hoses. Additionally, the fire main rupture model serves as a complement to the flooding module by

supplying information on the volume of water flowing out of ruptured pipes and contributing to flooding.

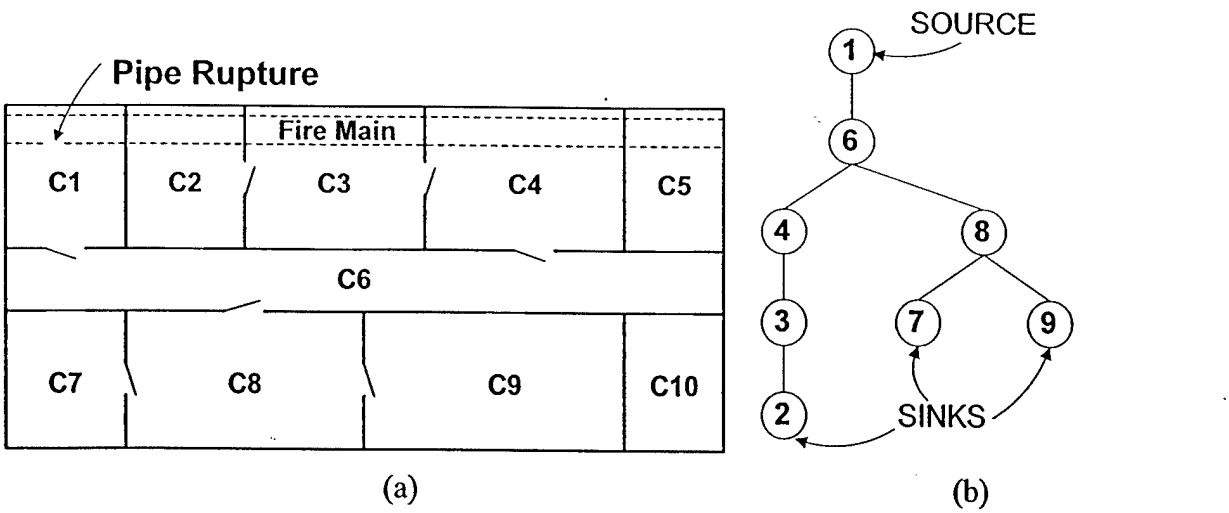


Figure 24. (a) A sample compartment configuration. (b) The corresponding flow network

Our approach to the modeling of both flooding and rupture relies on network flow methods. This requires the representation of the underlying systems by a graph, which is referred to as a flow network. Figures 24 and 25 show how flow networks are defined for the flooding and rupture models, respectively. There are three basic properties that govern the behavior of flow networks:

Capacity constraint: The net flow from one point to another may not exceed the given capacity constraint.

Skew symmetry: The net flow between two points in one direction is the negative of the net flow in the reverse direction.

Flow conservation: The net flow out of every point other than the source or secondary flooding is 0.

These properties permit the accurate modeling of water movement through a system by repeatedly propagating a given volume of water from one point to another along its flow path.

Both the flooding and rupture models use this technique. However, there are small differences. The flooding algorithm keeps track only of flooded compartments. It represents this collection of spaces by an undirected graph, where each node is a flooded compartment. The flow sources (those compartments containing a hull rupture or pipe rupture) are marked, as well as secondary flooding (those compartments that water enters from other ship spaces, but does not exit to other ship spaces). Water from each source is propagated to all reachable compartments by a breadth-first traversal of the graph, and, similarly for each secondary source, water is drawn off from all reachable compartments. Then, using the information so obtained, the principle of superposition is used to determine the net flow of water into or out of a compartment.

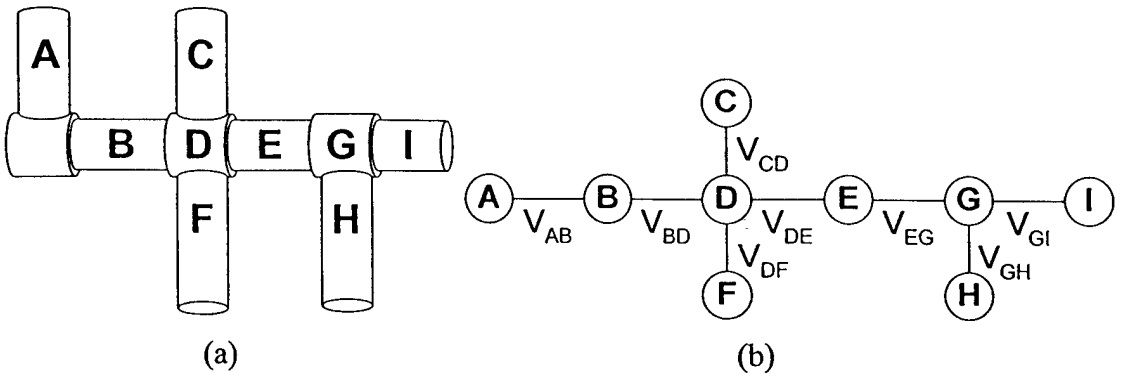


Figure 25. (a) A simple pipe system. (b) The corresponding flow network

The calculation of the actual volume of water transferred is accomplished via Bernoulli's equation (See Figure 26):

$$\frac{\Delta Q}{\Delta t} = A\sqrt{\frac{2(p1-p2)}{\rho}},$$

where $\Delta Q/\Delta t$ is the water flow rate, A is the area of the opening connecting the two spaces, $p1$ and $p2$ are the respective pressures in each space, and ρ is the density of the medium (64 lb/ft^3 for sea water). The flow rate is multiplied by the simulation time unit, Δt, to determine the actual volume of water transferred during that time between the compartments in question. In this fashion, water can be propagated from source compartments to all reachable compartments by repeated iteration of the algorithm in a breadth first manner.

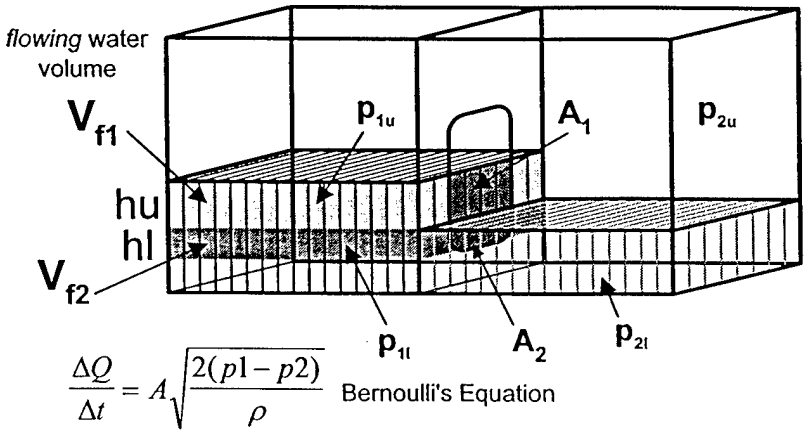


Figure 26. Illustration of the use of Bernoulli's equation to calculate the flow rate between compartments

The rupture algorithm works similarly to the flooding algorithm, except that the entire fire main system is represented by the undirected graph, where each node corresponds to a pipe segment. As in the flooding algorithm, source pipe segments (i.e., those directly connected to a fire pump) are marked, and sink pipe segments (i.e., those containing ruptures or having hoses or active sprinklers attached) are marked. Water is propagated from each source to all reachable pipe segments, and from all pipe segments to each reachable sink. Again, superposition is used to find the net flow, ΔQ, into or out of a pipe, which is then divided by

the time unit to obtain the flow rate ($\Delta Q/\Delta t$). Finally, for each pipe segment, the Bernoulli equation is applied in reverse to find the pressure change:

$$\Delta p = \frac{\left(\frac{\Delta Q}{\Delta t}\right)^2}{2A^2\rho}.$$

Thus, the rupture model can provide the supervisor with an accurate picture of the pressure at every point in the fire main. This information can assist him/her in making informed decisions on which ship systems can be activated at any given time, and which fire-fighting measures may be applied in a given situation.

3.3.5 Secondary Damage: Stability Module

When a ship takes a direct hit from an enemy mine or missile, stability is generally the most pressing and time-critical issue, since research shows that a ship is most likely to sink from foundering or capsizing within the first 15-20 minutes [Nemerich and Durkin, 1995]. Damage control officers must be able to make decisions as to what counter-flooding measures are to be taken to save the ship, or if this is impossible, to determine how much time remains before the ship is likely to sink. Currently, there is a manual procedure for making these calculations, but it takes upward of 40 minutes to complete the data collection and analysis alone. A viable software package that performs the analysis based on the SHCP model (FCCS) exists but still requires manual collection and entry of data. Neither approach produces results quickly enough to make them useful in a real crisis situation [Whitesel and Fairhead, 1996]. What is needed is a system that can automatically collect all necessary data from sensors located throughout the ship, perform the required analysis, and implement whatever actions are recommended through remote actuators.

The sensors required for input to a stability control system are flooding (liquid level), hatch and door closure status, fire main pressure and flow, and holing in the hull, decks, and bulkheads [Whitesel, et al., 1995]. None of these types of sensors are presently in use on a large scale or in the configuration necessary for damage control. However, some have been developed for and are utilized in civilian applications. These can be modified to adhere to military requirements. Others can be developed from scratch to fit the Navy's damage control specifications [Whitesel and Fairhead, 1996].

The actuators necessary for some stability issues are essentially those for opening and shutting fire main valves. They can be used to dynamically reconfigure the fire main (and other water-carrying systems) when ruptures in sections of the piping are detected. Fortunately, this type of actuator exists in current damage control systems. However, it is installed only on selected valves. To make the fire main damage control process completely automatic, actuators must be installed on every valve [Lestina, Runnerstrom, Davis, Durkin and Williams, 1999].

In addition to sensors and actuators, a reliable ship data network must exist for transmitting data between sensors/actuators and the computer system responsible for damage control. Navy ships currently utilize the Data Multiplex System to transmit electronic data between various modules. This communication system, however, is unreliable and cannot guarantee

uninterrupted transmission in a crisis situation [Whitesel and Fairhead, 1996]. Both hardware and software modifications must be implemented to make the network more robust.

Once sensor data are collected, they must be analyzed to determine the ship's current and projected levels of stability. The state-of-the-art package in use for this task is the Flooding Casualty and Control Software (FCCS), developed by NAVSEA 03H [Whitesel and Fairhead, 1996]. This package is considered reliable and is recommended by the US Navy for Fleet-wide implementation [Wujick and Rosborough, 1992].

Thus, a basic framework for shipboard stability control either exists or is already in advanced stages of development. The existing FCCS software could be used both in the Predictive Validation Module (see section 3.5.4) to help confirm sensor indications that a crisis is indeed in progress, and as part of the expert system that actually responds to a validated crisis. In the first case, the Predictive Validation Module would input real initial flooding, rupture, and structural damage sensor readings into the ship simulator (see section 3.3). The ship simulator would then model the progress of the incipient crisis (as indicated by the initial sensor readings) for several simulation cycles into the future. The simulated sensor reading thus obtained would then be input to the FCCS module to obtain an evaluation of the ship's stability some amount of time into the future. Based upon this information, a validation of the crisis would be obtained. Once it is known that a crisis is actually in progress, the FCCS module would be used to analyze the stability situation and recommend the response to maximize ship stability.

3.4 Subsystem 3: Total Ship Representation Subsystem

The Total Ship Representation Subsystem (Figure 27) contains the domain specific data that is used by all the other modules in the Supervisory Control system. It includes data and knowledge that is relevant to visualization and the system interfaces, to numerical ship simulation, and to intelligent reasoning. The basic format is an SQL relational database. All modules access the relational database. They also update the database when their behavior changes the state of the world. One of the four PCs of the Supervisory Control system is devoted to total ship representation. It is called the Knowledge Base PC.

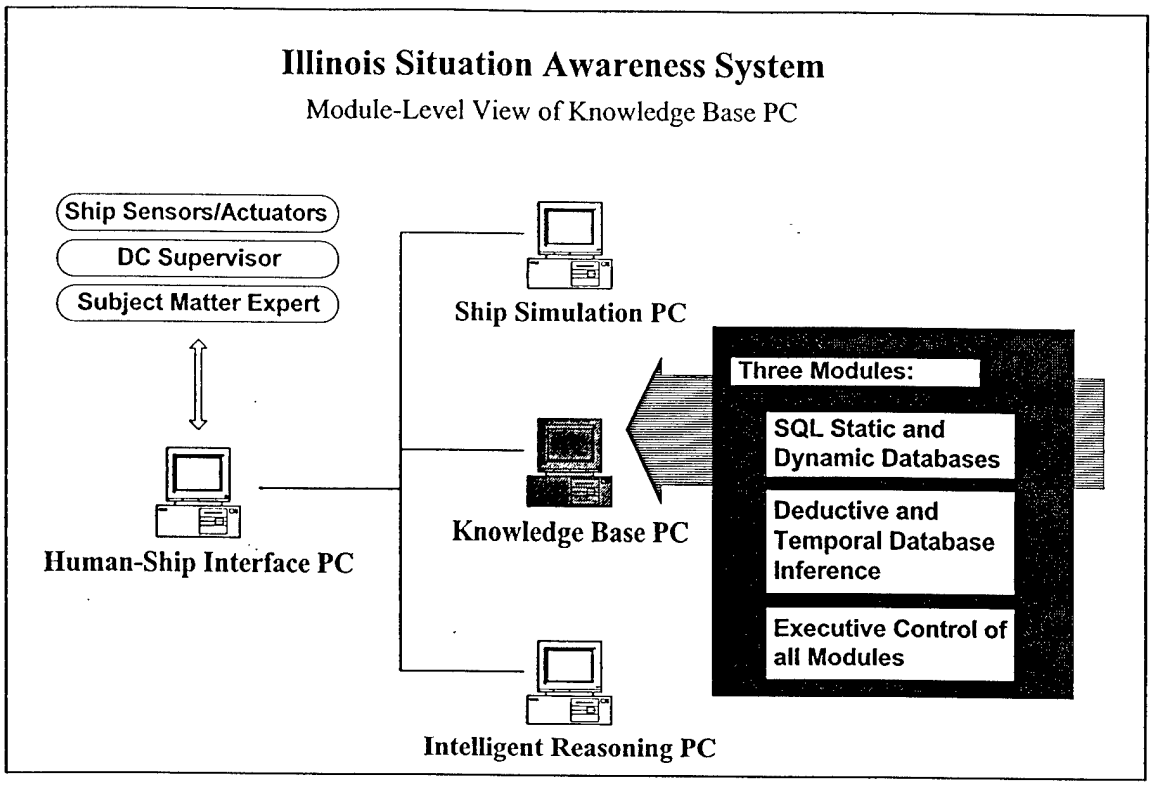


Figure 27. The Total Ship Representation Subsystem

This section deals with the formal representation of the domain. Any successful AI project has to be built on top of a solid domain representation. Such a representation is the foundation of the entire system and provides:

- A formalization to reason over
- A framework for dynamic and static data storage
- A common format to facilitate inter-module communication and data exchange.

In our approach the representation could be divided in several areas:

- Static structures and data
- Dynamic structures and data (including temporal reasoning data)
- Executive control

The knowledge base is implemented in an industry-standard SQL-compatible database that is accessed through the Microsoft Windows Open Database Connectivity (ODBC) interface.

3.4.1 SQL Database Module

The SQL Database module is the central knowledge repository for the entire system. All other modules interface only to it to retrieve and report information about the ship. The database maintains all static, dynamic and temporal information.

The knowledge base ontology is realized as a relational, SQL-compatible database. All modules access the database using the Windows ODBC interface, which acts as a layer between the database and the client application. This allows extensive flexibility in the

implementation of the other modules. For example, the modules can be implemented in Microsoft Visual C++, Borland Delphi, Lisp, Prolog, FORTRAN, and specialized expert system shells.

The design of the database makes it easy to specify different versions of ships, and the data associated with a crisis scenario can be saved for later analysis.

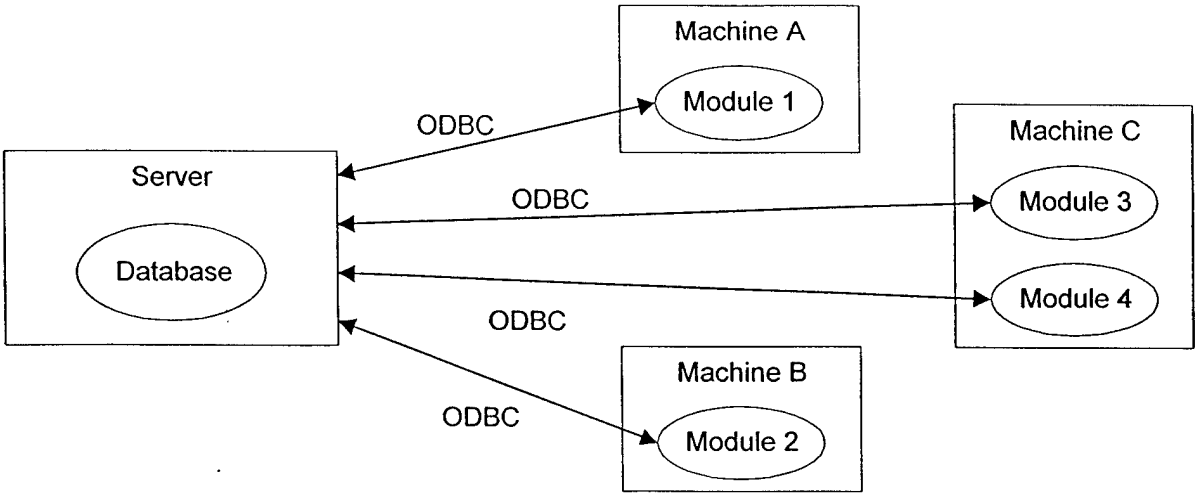


Figure 28. The knowledge base access topology

Shown in Figure 28, the modules can run on any configuration of network machines. The modules could each run on a separate machine, or they could all run on the same machine. As long as there is an ODBC link (established when the network is set up) to connect them to the central database, the modules will operate the same regardless of the network topology. As discussed below, it is also possible to distribute the database to improve data access efficiency instead of keeping it on a single server.

The knowledge base is divided by ship class into several broad categories:

- Status of compartments
- Status of fire main, chilled water and other relevant shipboard systems
- Status of ship stations (such as repair lockers and medical)

For example, the database contains a table called CW_Valve_SHIP, which lists static data about the location of chilled water valves by associating a valve number with a compartment specification (Number stands for the valve number):

CW_Valve_SHIP fields:
Deck
Frame
Position
Number

Another example of a static table is Station_SHIP, which lists information about personnel stations (such as a repair locker):

Station_SHIP fields:
Station_code
Station_description
Personnel

The following figure illustrates how the SHIP AutoCAD ship data are represented in related tables in the knowledge base:

Information that evolves over the course of a crisis is maintained in dynamic tables, such as FW_Pump_Status_SHIP. A set of sample records is given below, showing how a sequence of changes in the status of a particular pump is recorded (Cell stands for compartment subdivision):

Deck	Frame	Position	Cell	Number	Status	Time
3	370	2	1	2	2	14:32.34
3	370	2	1	2	1	14:35.01
3	370	2	1	2	0	14:35.22

All of the data usually contained in a single AutoCAD file are distributed into many tables in the knowledge base, where there exists a table for each type of object on the ship such as compartments, water system components and personnel stations and table(s) identifying the relationships among them, as demonstrated in Figure 29. Making entities and relationships among them as explicit as possible simplifies the process of reasoning about them in an AI environment. A "1-to-N" link between tables signifies that for the indicated field in the first table, there exists one record with a particular value in that field and in the second table, there exists one or more records with that value in that field. This linking of tables via related data is at the core of relational databases. The reason that "flattening out" the AutoCAD data into a database is preferable to simply using the AutoCAD file is because the database stores data in a way that is amenable to AI reasoning and modification.

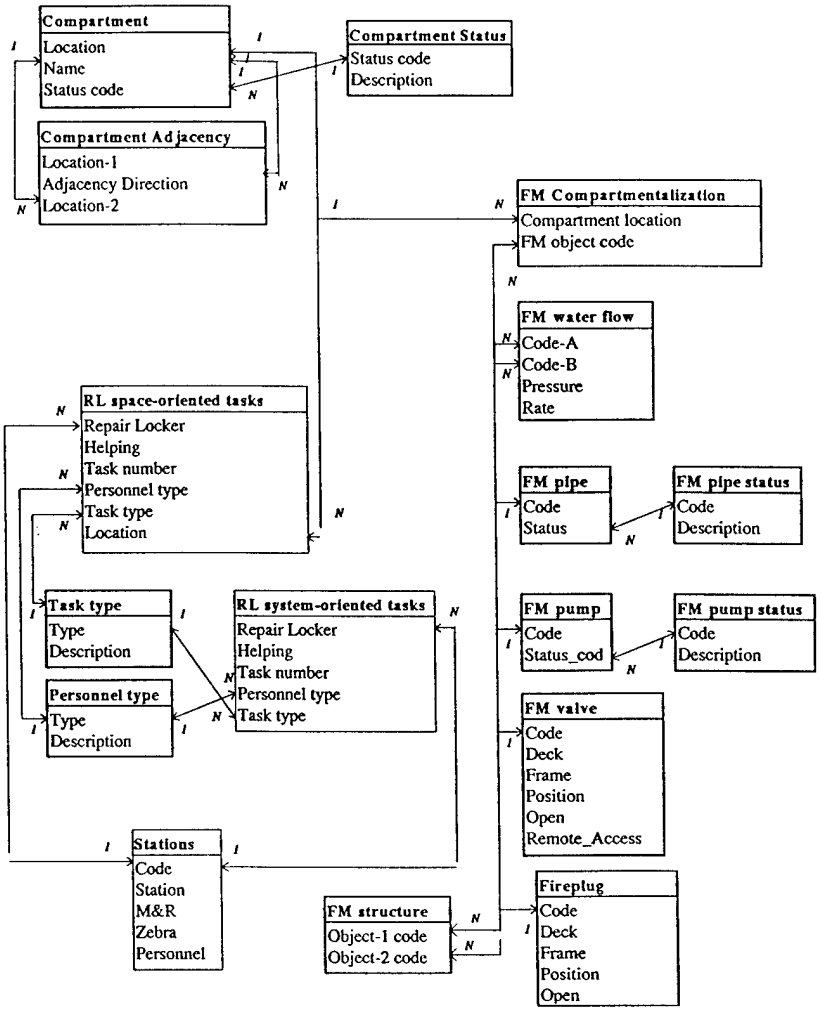


Figure 29. A portion of the database schema that defines part of the knowledge base

All dynamic tables are a log of events ordered by the time at which they occurred – in this case, we see that pump #2 in compartment 3-370-2 changed from status 2 to 1 to 0 at the indicated times. This keeps a record of everything that happens during a crisis and allows modules to analyze temporal relations among events.

It is desirable to implement the knowledge base using proven and reliable commercial off-the-shelf (COTS) technology as much as possible to avoid introducing proprietary formats. Currently, Microsoft Access is the Database Management System being used. However, other commercial formats could be accommodated transparently since Windows' ODBC layer hides the implementation details of a particular database from the client application. For example, if Microsoft Access does not provide sufficient performance we consider using Microsoft SQL Server. The only requirement is that the database use the standard relational database paradigm, which is why, for example, the AutoCAD data of the ship structure was "flattened" into a set of Access tables. This ensures maximum portability and compatibility.

A major consideration in designing the knowledge base implementation is speed. We can expect a great deal of read/write accesses to the dynamic tables during a crisis.

Current benchmarks indicate that a 200Mhz Pentium Pro system can sustain approximately 6000 reads/second and 600 writes/second to a single table. Performance decreases when several processes (possibly on different network machines) access the same table. To guarantee that large amounts of information can be delivered promptly in a time-critical situation, a high-speed local area network (LAN), such as 100Mbps Fast Ethernet, and low-latency, high-throughput hard drives are required. This technology is readily available on the commercial market. Additionally, each machine on the LAN can store a copy of the knowledge base on its own hard drive, allowing the modules running on that machine to access a local file instead of sending a data request over the network. Only the global changes made to the knowledge base are propagated over the network to the other machines instead of every data request, reducing network load.

The knowledge base is also used for inter-module communication, in keeping with the notion that it is the "glue" that binds the system together.

3.4.2 Deductive and Temporal Reasoning Module

The Microsoft SQL database is a relational database, and hence all the information is at a ground level -- the level of tables. The deductive and temporal reasoning module allows more complex inferences to be drawn by reasoning over the SQL module.

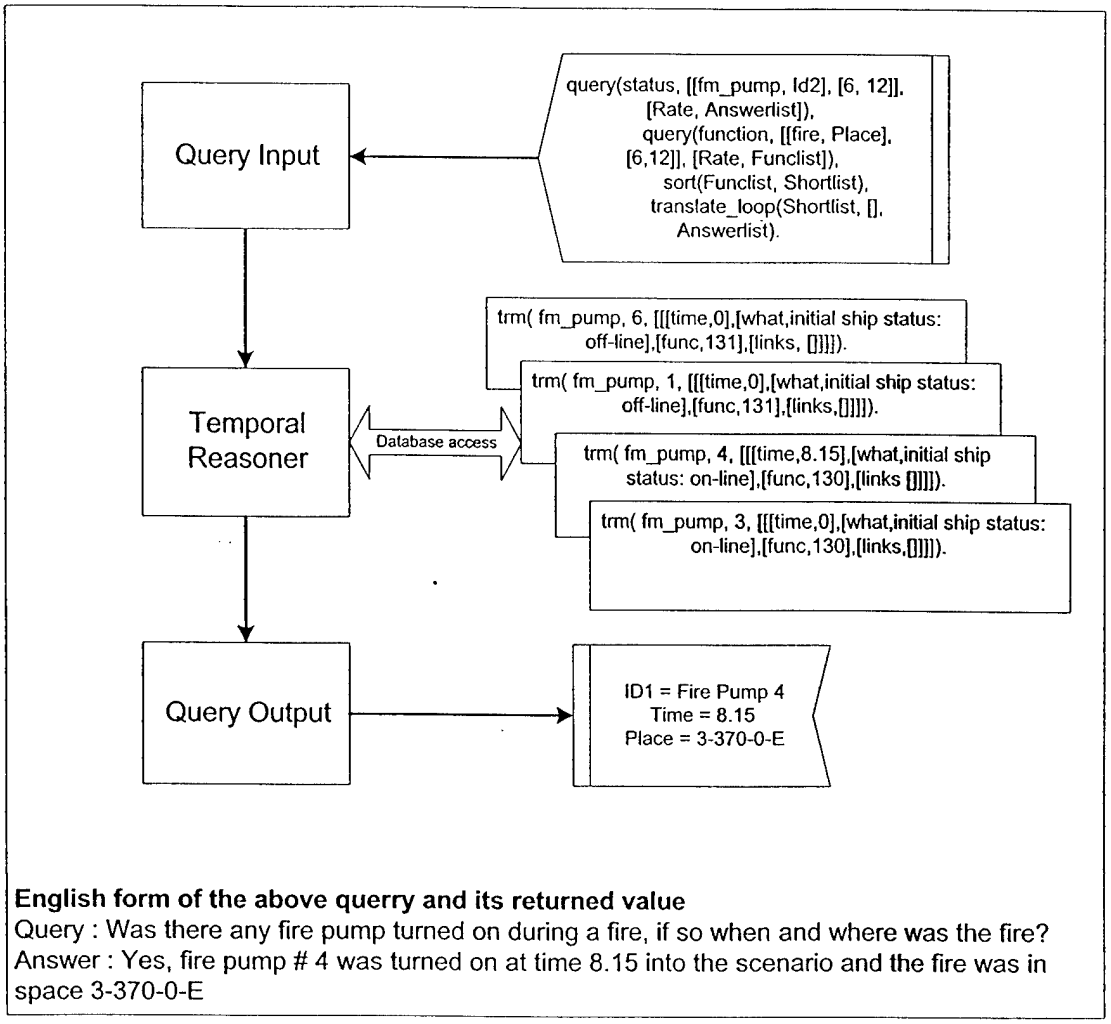
The damage control (DC) domain is one of time-critical and resource-constrained decision-making. Time-critical in the sense that proliferating crises could damage surrounding areas and vital systems, and resource-constrained because on a ship, there is only a limited number of manual and automatic equipment available to deal with crises.

Temporal models can be classified according to their *primitives of time* [EQUATOR, 1990]. In *point-based* theories the representation deals with individual points of time corresponding to start and end times for events. In such theories reasoning is achieved over precedence, successions and simultaneity, and their negations and combinations. On the other hand, *segment-based* (or *interval-based*) theories represent intervals of time that correspond to the duration of the event. Allen [1983] introduced a temporal logic that is represented as distinct time intervals and the various associations between them. An interval can be described as a discrete segment of time used to stipulate a period of time over which an action or event holds. As there can be several intervals on a time line, Allen also introduced some common relationships and reasoning over them that we will represent in our model.

In Dean & McDermott, [1987] the authors examine components of temporal databases in terms of *time map maintenance* (TMM), partially specified constraints and resolution strategies for conflicts in the database using *clipping*. Shoham and Goyal [1987] use time intervals and assertions about them that allow the significant objective of being able to interpret over intervals of *zero* length. Further, their representation uses a pair of end-points for intervals of time that are considered *linear* and *dense*, and they do not assume *homogeneity* in the truth of the assumptions over sub-intervals. The automatic referencing over interval hierarchies in the TIMELOGIC [Koomen, 1989a] system has a strong correlation to the representation issues being tackled here.

3.4.2.1 The Temporal Resource Manager

The temporal resource manager (Figure 30) for the DC domain is based on the principles of a temporal database [Dean & McDermott, 1987; Maiocchi, 1988; Shoham, 1994]. Information is stored in the database according to time intervals based on Allen's time interval work, as well as categorized into a classification of what aspects of the ship the particular piece of information affects. The Temporal Resource Manager (TRM) can be broken down into a series of tables. Each table contains information about its specific domain on the ship. For each system action (command input such as 'turn on firepump #2') into the TRM, an entry is made into one of these tables. Table entries are not disjoint; one command often updates three or more tables. Table domains often overlap and many commands depending on the variables relate to many different domains and could be a subset of a larger table. The redundancy in tables is used to aid in consistency and speed when retrieving information. There are over 120 different commands that exist in the current version of the TRM. Each individual command contains a unique variable list to specify the exact command, although for similar commands an attempt was made for some degree of variable uniformity. Every variable has a number of different possible values recognized by the system depending on the variable and the command associated with it, so there are a finite number of commands that can be input into the TRM. Each command need not use all of its variables. Each table entry contains the command, the time at which the information was reported, and any variables associated with the command. The variables are internally stored in a table format with each table consisting of an identifier for the variable and the value. In this way, the order of the variables need not be consistent, or even the variables associated with each command. The table entries are stored chronologically thus making the task of finding a specific *crisis interval* trivial just by iterating through the table entries, as not all entries in a table may specifically refer to a given interval.



English form of the above querry and its returned value
Query : Was there any fire pump turned on during a fire, if so when and where was the fire?
Answer : Yes, fire pump # 4 was turned on at time 8.15 into the scenario and the fire was in space 3-370-0-E

Figure 30. The above Figure shows the power that is gained by using a temporal reasoner that reasons over the contents of the database to answer intelligent high-level queries

The tables are also broken up into various categories, each of which is concerned with a specific division of the ship. Every major piece of machinery on the ship has a table associated with it. Examples of crisis tables are those associated with a fire in a specific compartment or a rupture in part of the firemain. Although every entry in a crisis table is also duplicated in another table and thus a crisis table could actually be constructed from the other tables, speed and simplicity are at the cost of redundancy. Crisis tables are created dynamically. Each different TRM command has a data rule associated with it. This rule specifies which set of tables will be updated and what specific information will be stored in those tables. For different variants of a command, there may be multiple rule associations. When multiple rules are present, the correct rule is chosen by a meta-rule type associated with a command.

The initial report of a fire (say at time i) creates a time interval I_{ij}, which denotes the events, related to the fire (here the end of the interval is left unspecified since it may ultimately lead to the sinking of the ship). The interval can be viewed as a bound for certain activities that can be associated with fighting the fire. The individual events that constitute this time

interval can either be specified as a total order of events/actions that have to occur within the interval frame or partially ordered or as an incomplete order. This theory seems consistent (to some extent) with the findings of Song [1992] and Koomen [1989b]. Now another interval (or event) E_k corresponding to the starting of a fire pump was asserted with a report from the DCCO at time k. In order to ascertain the effectiveness of the DCA's action we need to know the temporal relationship between I_{ij} and E_k. Thus the TRM can be used by the expert systems that control and regulate crisis management activities on the ship by supplying information in a temporal and context sensitive manner. This will allow the expert system to reason over a set of events that occurred over an interval of time and its relationship to other temporal events.

3.4.3 Static and Dynamic Database Interface Module

This module defines the entity-relationship schemas of any relevant, preexisting static or dynamic databases. These schemas allow the intelligent reasoning system to use these databases for intelligent reasoning. It is assumed that all of the static and dynamic databases support the ODBC interface convention.

Because all modules are designed to interface through the industry-standard ODBC layer, the knowledge base can be expanded indefinitely with external databases provided by the Navy and its contractors. The only requirement is that these databases be in a common database format such as Access, SQL Server, Oracle, Dbase or Sybase. Upon receipt of a new database, the knowledge base ontology is simply extended to accommodate the new data set; this allows the DC-ARM modules to harness the information contained in those data sets with minimal additional effort.

The static and dynamic database interface module is, then, a gateway from all the required databases to the modules that use them during a crisis. Any database that the Navy is able to provide could be easily integrated into the DC-ARM system.

3.4.4 Executive Control Module

The executive control module (Figure 31) will be responsible for invoking other modules as needed, coordinating their activities, and identifying and responding to failures within the system. Our principal design goals are to allow flexibility of module implementation, uniformity of interface between modules, optimum performance, and maximum survivability. The core architecture will be a master process on the core server that will use the Windows NT Remote Procedure Call (RPC) mechanism to invoke instances of the other modules on any machine in the system and pass messages and notice of critical events between modules. Each module will be responsible for "checking in" with the executive control module at specified intervals to report current status. The executive control module will be responsible for investigating modules that fail to check in or report error conditions, and restarting them as necessary if they have crashed or stalled. In addition, the executive control module will be responsible for monitoring system performance on each machine and attempting to balance workloads between machines, as well as detecting and responding to hardware failures within the system (e.g. broken network connections or systems going off-line.)

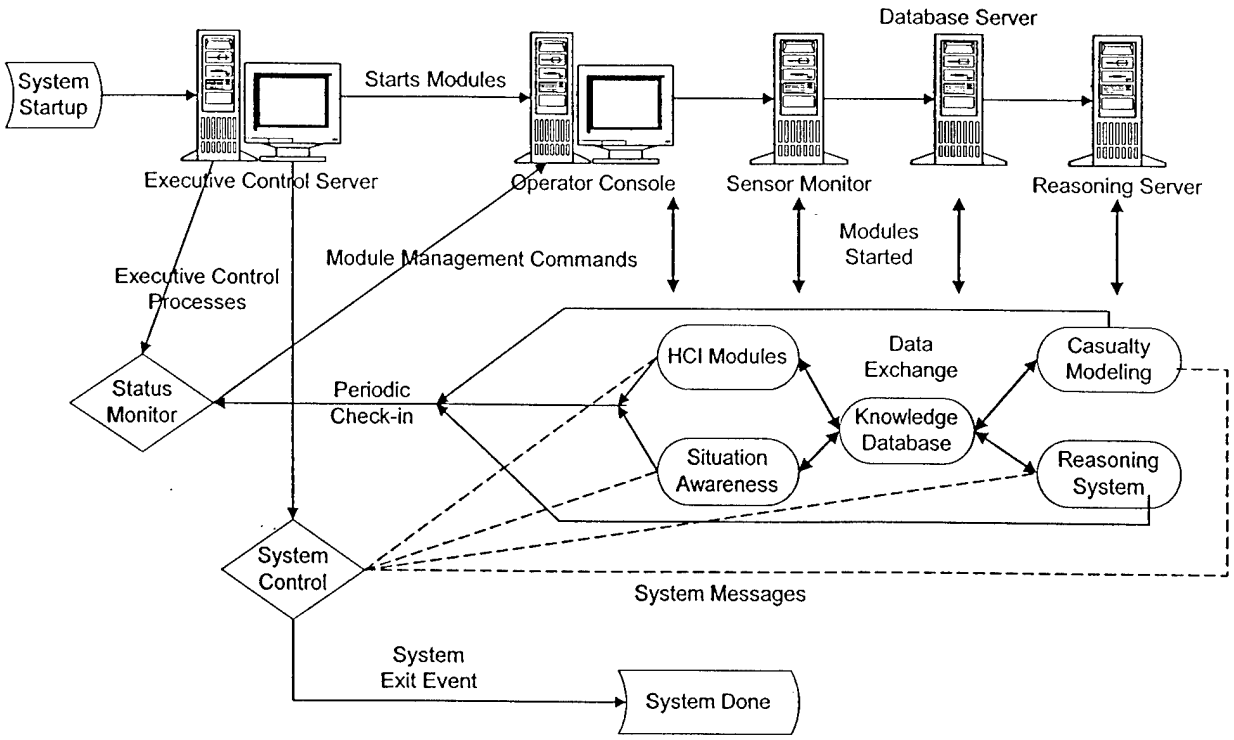


Figure 31. An illustration of how the executive control module coordinates the various processes

Whenever possible, modules will be designed with multiple threads of execution and parallel processing in mind, so that the executive control module can invoke or kill instances of each module as necessary and better manage system load balance between machines. For example, the smoke propagation module might be invoked once for each new smoke-causing casualty that is detected and have a separate thread of execution processing each instance. This allows dynamic response to events and has the advantage that it does not require an instance of rarely used modules to be active at all times, continuously polling the global event database to see if new events have occurred which interest them, greatly improving system response times. In the case of modules which naturally have information to process at all times, such as the operator interface manager, this paradigm does not have to be adopted and the originally invoked instance of the module can remain active throughout the system's activity cycle.

Damage control systems naturally must operate in unstable environments with a very real possibility of capability loss due to casualty, and ensuring that the entire system is as fault-tolerant and resilient as possible is the primary responsibility of the executive control module. It will be designed specifically to make no assumptions about the status of system components, and to detect and respond to losses of functionality as rapidly and flexibly as possible. Additionally, it will be responsible for monitoring inter-module communication and ensuring that all requests for information are responded to within a reasonable amount of time to guarantee that no module will stall waiting for a response from another module that has crashed, and reporting significant error conditions to appropriate modules.

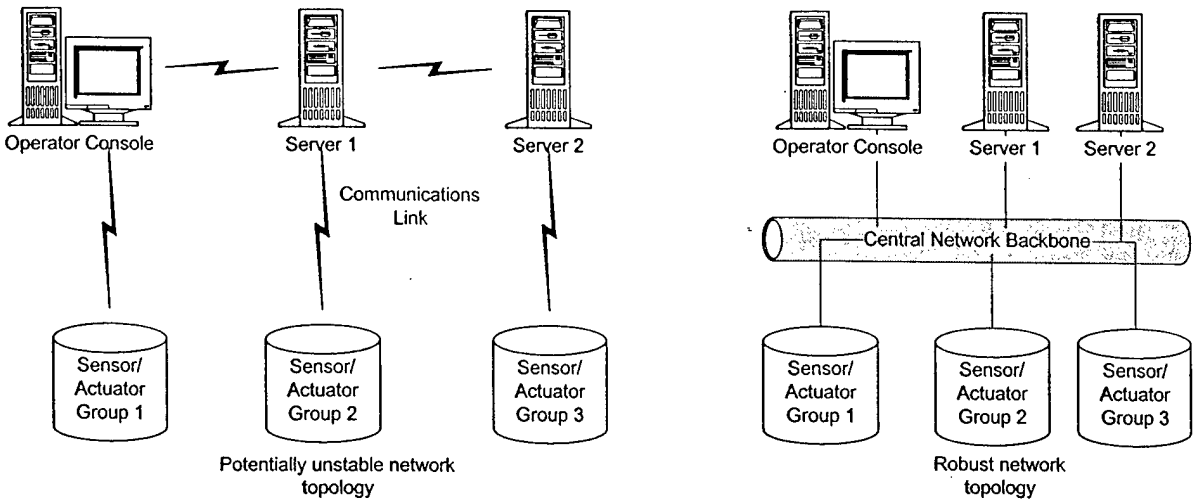


Figure 32. Comparison of potential network architectures

The machine breakdown given in Figure 32 is virtual rather than physical; it is convenient to think of the collective resources required by each module and system of modules as a separate entity, but it is our intention to present our hardware collection as a single resource pool to the executive control system, so that no single computer is reserved exclusively for any specific set of tasks, and any instance of any module might be run on any machine in the cluster. This flexibility will even be extended to the executive control module itself, so that if another module detects that the executive control server is off-line it can restart the executive control module on its own machine in a special emergency state, from which the executive control module will assess current system status and take over management duties. This will obviate the need for reliance on a certain core server to remain functional at all times, and allow the system to better withstand losses in component functionality due to ship damage or hardware/software failure. By interconnecting system hardware with a topology that ensures that a broken link between any two components does not isolate functional subsets of the system from each other, this architecture will require that all computational resources or connections are completely destroyed before it is rendered impotent, a key requirement of a system as critical as damage control management.

3.5 Subsystem 4: Intelligent Reasoning Subsystem

This section describes the intelligent reasoning subsystem (Figure 33). The intelligent reasoning system consists of the Supervisory Control expert system, the machine learning system, the crisis predictive validation system, and the intelligent object system. The blackboard expert systems for crisis recognition and casualty response are at the core of this report. Their functions are to estimate the present world state based on all sensory information, evaluate the possible courses of action, and select the action with the highest expected value. All of this has to be done within a noisy real-time environment and within a large state-space.

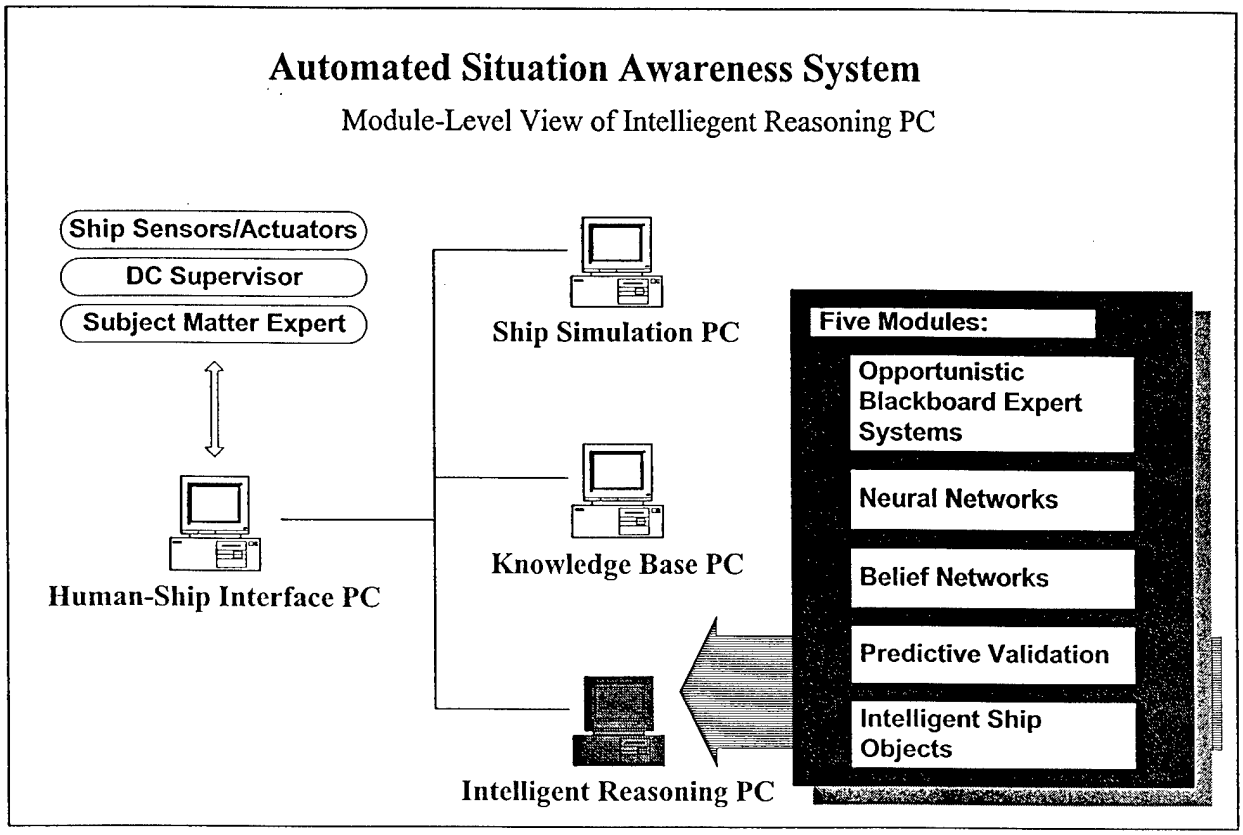


Figure 33. Module-level view of the intelligent reasoning subsystem

3.5.1 Blackboard Expert Systems

The Blackboard Expert System subsection contains discussions about its relationship to the other parts of the intelligent reasoning system, and the research contributions we expect to make during the course of the project. Because of the difficulty of the task and the lack of expertise on performing damage control on future ship designs, which can be created within our intended architecture, there is a natural place for a machine-learning module. The machine-learning module consists of two parts, a sub symbolic (artificial neural networks)

and a symbolic (rule-based learning and decision network learning). We outline below the role which both of these reasoning and learning paradigms may play in a damage control context and present some central research challenges and how we expect to address them within the Automated Supervisory Control System. The crisis predictive validation system is concerned with crisis diagnosis using forward simulation, as is the intelligent object system, which will initially provide simulated behavior of sensors to the Supervisory Control expert system. Later, these intelligent objects will serve as an interface between the expert system and the real sensors

3.5.1.1 Challenges for the Blackboard Expert Systems

One of the largest challenges of the project is crisis recognition and response. The blackboard expert systems are the heart of the entire project and are exposed to many challenges including the following:

- Real-time reliable crisis recognition
- Crisis recognition not explicitly programmed
- Real-time response
- Need for dynamic knowledge refinement
- Need for learning from massive pool of examples (hundreds of thousands)
- Integration of static and dynamic databases
- Integration of various nature reasoning subsystems (Bayesian networks, artificial neural nets, rule-based and case-based systems
- Need for learning and knowledge refinement for most of those subsystems
- Integration of the simulator into the expert system
- Integration of the supervisor "override" mode into the expert system
- Need for sensor fusion
- Need for certain domain-independence (working from the first principles)

3.5.1.2 Background on Blackboard Architectures

Some of the most successful Artificial Intelligence applications are expert systems. In turn some of the most successful expert systems have been built using blackboard architecture. Blackboard architecture is suitable for ill-defined and complex domains [Nii, 1989; Simon, 1969; Newell, 1969]. Clearly, the damage control domain falls within this purview. Classical examples of successful blackboard expert systems are HEARSAY, HEARSAY-III [Reddy et al., 1973; Erman et al., 1981] as well as the HASP system [Nii et al., 1978]. The latter one is an especially valuable example as its task was platform recognition from sonar array readings. Like our intended system, HASP used sensor fusion and multi-level knowledge representation.

Recently, blackboard systems have been refined in various ways. For instance, classical scheduler has been improved by means of recursive classification [Park, Donoho, and Wilkins, 1991; Wilkins, 1996]. A particularly interesting recent project, called Guardian,

was developed at the Knowledge System Laboratory at Stanford University [Hayes-Roth et al., 1992; Hayes-Roth, 1994a, 1994b]. The system deals with handling patients in an intensive care unit. In many senses the task of Guardian is analogous to the one of our system, as both domains require real-time crisis management and situation monitoring and both domains are sufficiently complex.

We plan to build upon the past Guardian work in blackboard architectures [Hayes-Roth et al., 1992] as well as our own work on blackboard architecture systems [Park, Donoho, Tan, Mengshoel, & Wilkins, 1995; Wilkins, 1996], since Guardian was intended as a "proof of concept" only.

3.5.1.3 Overall Design of the Supervisory Control System

We propose to develop a novel multi-blackboard based expert system capable of dealing with challenges imposed by the damage control domain. A high-level structure of the system is depicted in Figure 34, which was also shown earlier in the report.

3.5.1.4 Blackboards

The two main blackboard systems for Supervisory Control handle the tasks of crisis recognition and casualty response. The crisis recognition system takes as input smart sensor data as well as information from static and dynamic ship databases. Its output is fed into the second expert system, which deals with the subtasks of casualty response.

Figure 35 illustrates some of the main elements of a blackboard system.

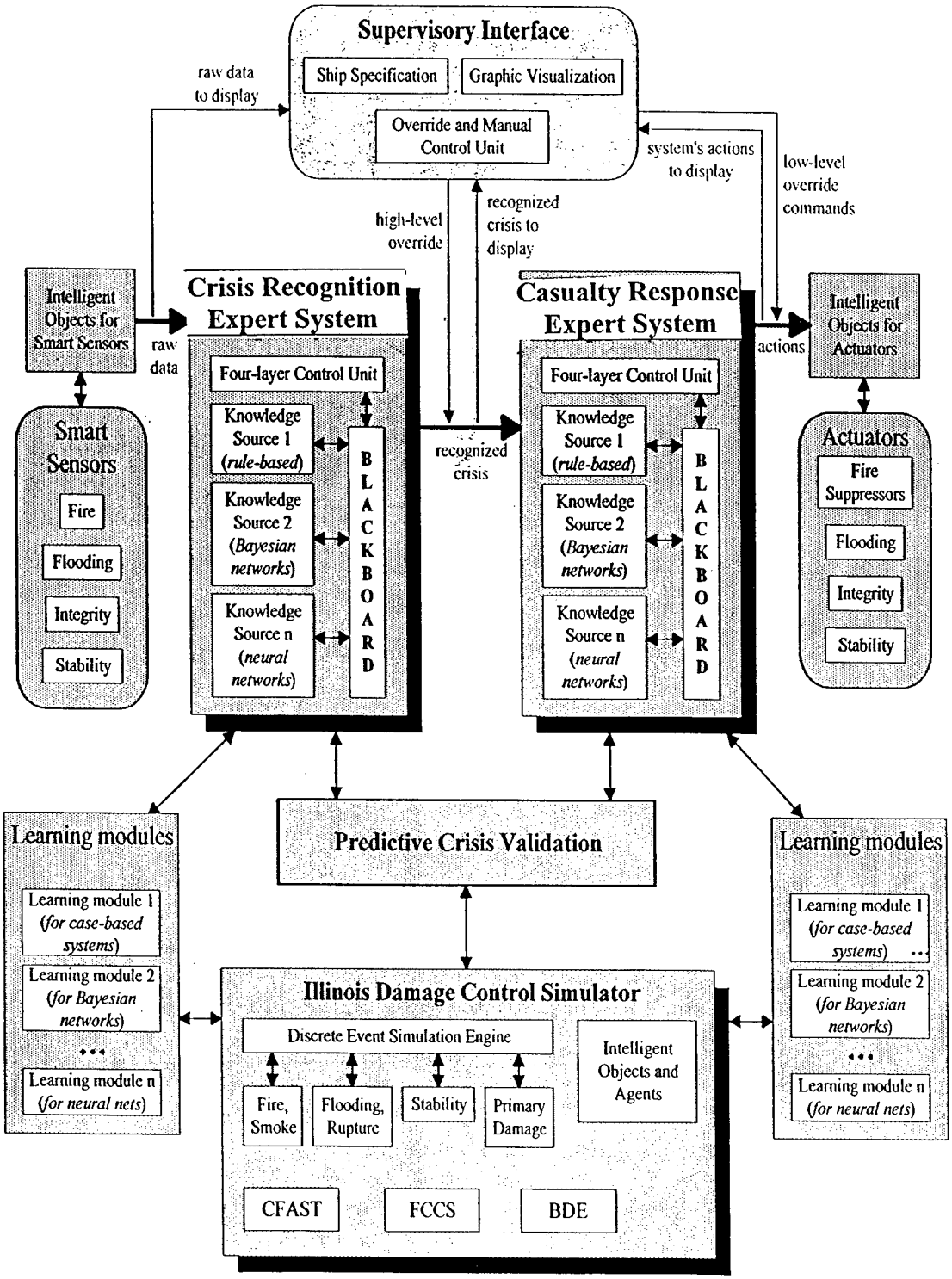


Figure 34. Functional Overview of the Illinois Automated Supervisory Control System

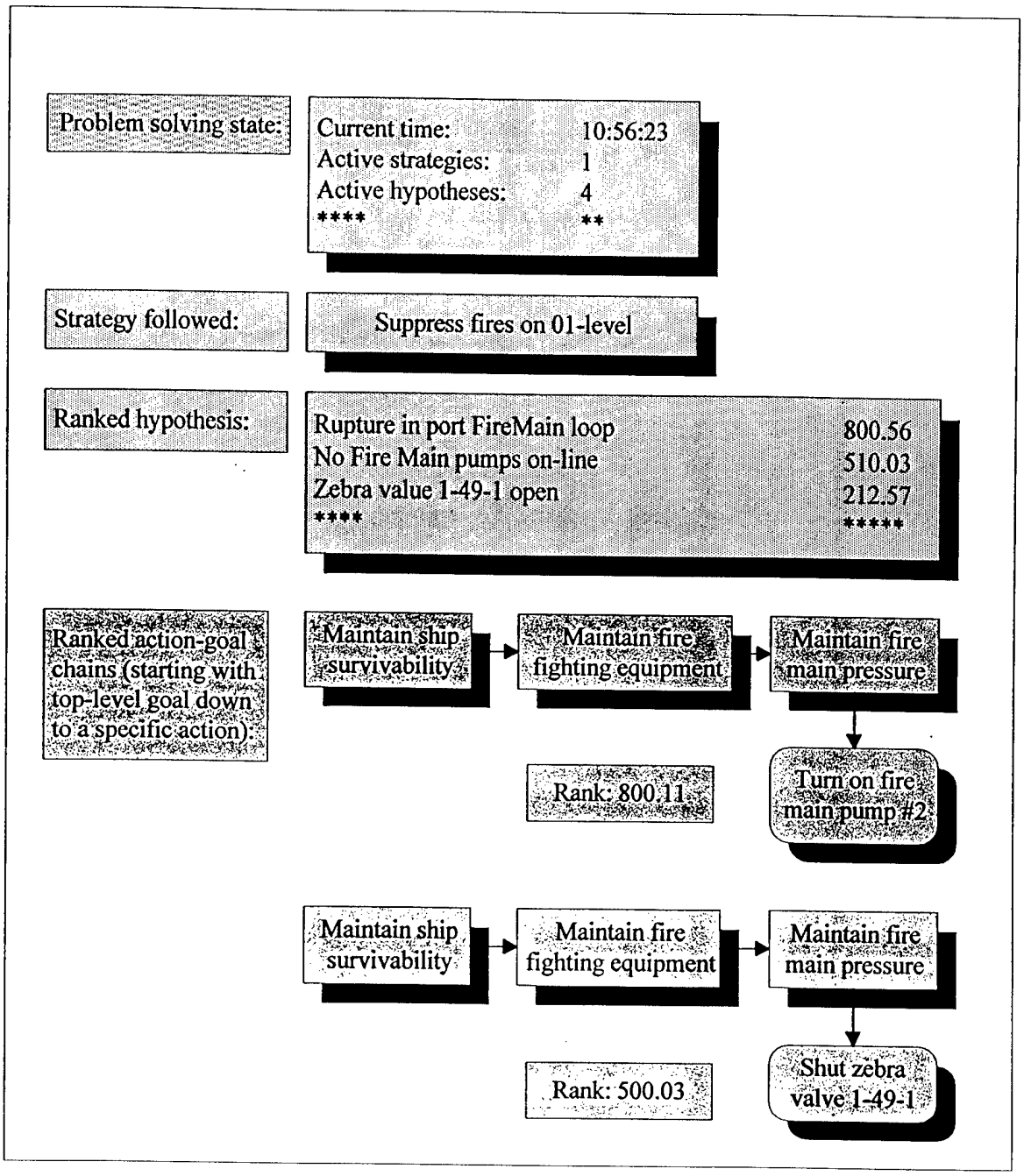


Figure 35. Blackboard items

3.5.1.5 Knowledge Sources

Like any opportunistic system the blackboard expert system contains a number of knowledge sources. The knowledge sources could have different natures and therefore allow for different types of reasoning. The knowledge sources monitor the blackboard and read the data on it. As soon as a knowledge source realizes that it can somehow participate in the problem-solving process (i.e. the preconditions are satisfied) it takes the information

necessary from the blackboard, processes it, and posts the results back on the blackboard. The results could be of the following types:

- New hypothesis
- New datum
- Request for more information
- Suggested action

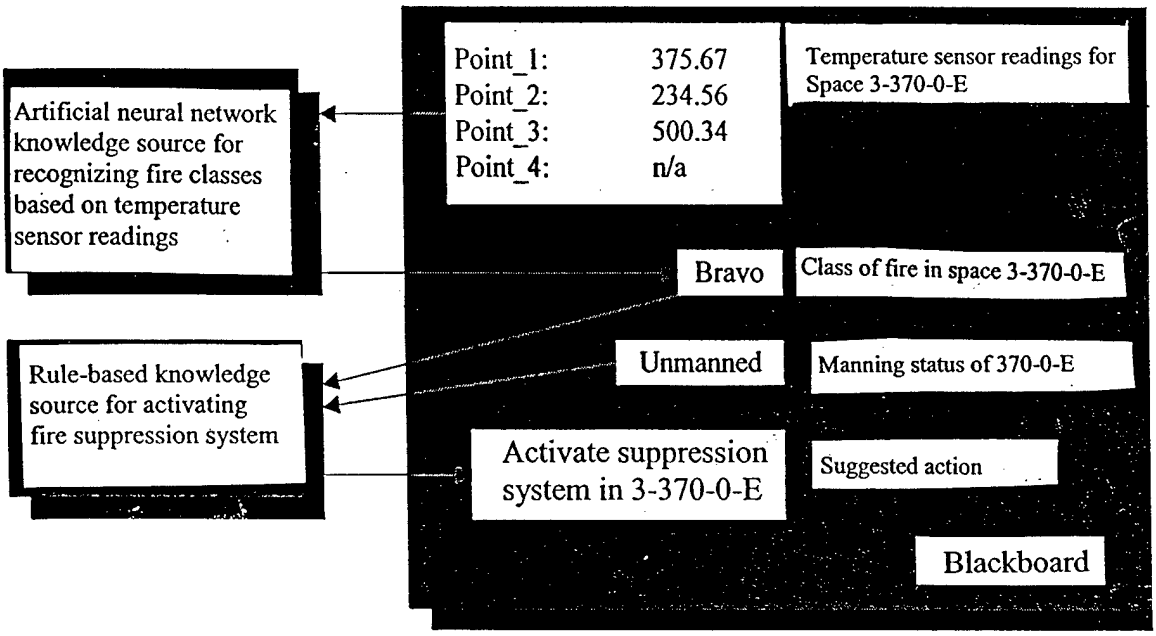


Figure 36. Knowledge source interaction in the problem-solving process

Figure 36 is a simple example of the collaboration of two sources of knowledge. In this example, the neural network knowledge source, belonging to the crisis recognition expert system, picks up the temperature readings from four sensors located in different points of compartment 3-370-0-E. The output of the network is the fire class (bravo in this case). This datum gets posted on the causal response expert system blackboard where it gets picked up by another knowledge source. Having a strict procedure for activating the suppression system and given the type of the space (unmanned), the knowledge source decides to activate the halon system in the compartment and posts it on the blackboard as a suggested action. This example clearly shows that we can compensate for, and even take advantage of, having *non-uniform* domain theory. In other words, some aspects of the domain (e.g. setting Zebra) have a strict procedure behind them while others (such as deciding whether to flood VLS compartment) do not. Therefore we can utilize strong domain theory based methods (e.g. rule-based) for building knowledge sources dealing with the aspects of the first kind while making use of non-symbolic methods (like artificial neural networks) to deal with aspects of the second type.

3.5.1.6 Control Unit

The SA expert system also contains the Four-layer Control Units. Each unit manages the entire problem solving process on corresponding blackboard and works in cycles as shown in Figure 37.

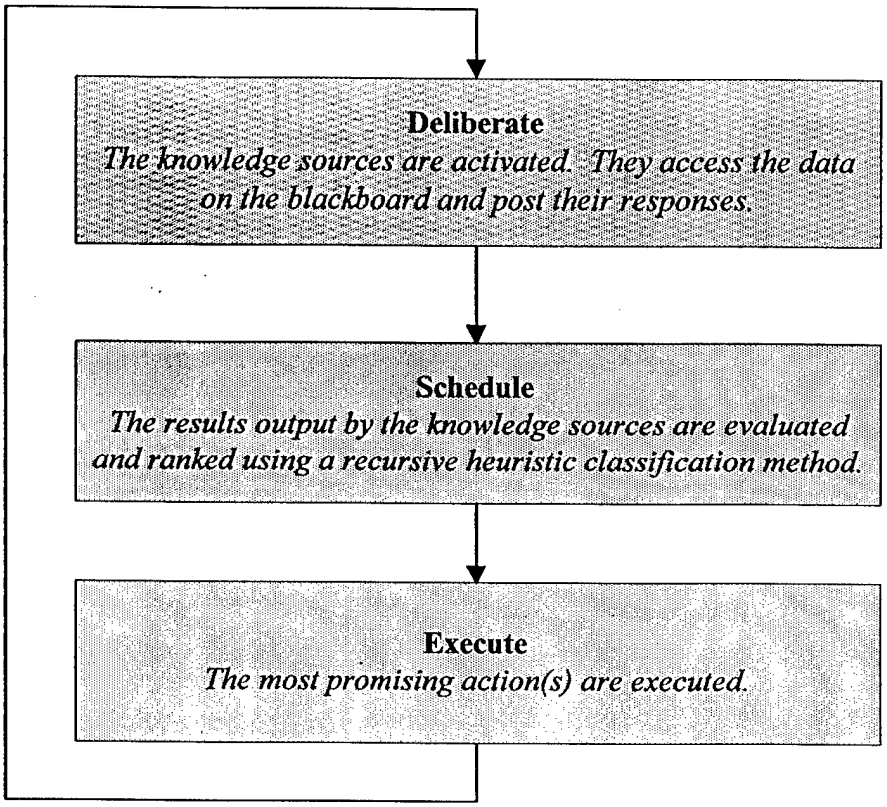


Figure 37. Blackboard Control Cycle

We use an advanced recursive classification scheduler [Park, Donoho, and Wilkins, 1993]. One of the evaluation techniques it uses is reasoning over the action-goal chains. It allows for domain-independent concept implementation (e.g. "rule out a hypothesis").

3.5.1.7 Simulator

One of the novel aspects of our approach is using the simulator (Section 3.3 of the report). The simulator is capable of real-time fire, smoke, flooding, rupture, and stability simulation. It uses discrete event as well as intelligent object/agents based architectures. There are two uses of the simulator. First, it will be used to train the expert systems on hundreds of thousands of various damage control examples. The learning module will provide a necessary interface for that. This simulator use is of high importance to the entire system as it is unfeasible to supply the system with a sufficient number of actual training examples (taken from ex-USS *Shadwell*, for example). The second use of the simulator is the run-time predictive validation (Section 3.5.4). As is well known, early crisis recognition is somewhat

difficult because of noisy sensor data as well as false alarms. Our design deals with that aspect by running real and hyper-real-time look-ahead simulation. The results of the simulation are compared with new sensor readings for crisis validation.

3.5.1.8 Smart Sensors and Actuators

The SA expert system will be hooked up to ship smart sensors and actuators. Intelligent objects for smart sensors and actuators will serve as an interface layer between the expert system and actual sensors/actuators. They will be capable of doing low-level information preprocessing. Another use of the intelligent objects is simulation of the physical sensors and actuators before the latter are implemented.

3.5.1.9 Supervisory Interface

The supervisory interface (described in Section 3.2.4) provides interaction between a human expert and an expert system. Its primary functions are:

Ship visualization module (Section 3.2.1) to display real-time crisis management information using an immersive multimedia interface.

Override and manual control unit to provide human supervisors with the opportunity for real-time rapid system interruption and manual control (as described in Section 3.2.4). The research goal here is to achieve intelligent interruption so the system is aware of human interruption happening and does not struggle with human experts trying to regain control. We implement it with multi-level supervisory interface inputs and outputs. As the diagram at the beginning of Section 3.5.1 shows the supervisory interface unit has:

- two inputs – low-level data from the intelligent objects for smart sensors, and high-level data from the crisis recognition expert system;
- two outputs – low-level override commands fed to intelligent objects for actuators, and high-level override commands fed to causal response expert system.

The inputs supply low and high level information necessary for human experts to supervise damage control on a ship. The outputs allow for intelligent override by overriding both inputs and outputs of the causal response expert system. That gives a lot of flexibility in human-computer damage control cooperation ranging from automatic mode to completely manual mode.

Ship and scenario specification units (sections 3.2.2 and 3.2.3) allow for crisis specification including ship configuration and multiple crisis initiations. The unit will be used in process of initial and incremental system training.

3.5.1.10 Learning Module

The last component of the design, in order of discussion, is the learning module (Section 3.5.2). The module includes sub-modules for different nature knowledge sources learning. Some of the expert system's knowledge sources are hard-coded and don't require learning. This situation occurs primarily when the domain sub-theory is strong enough; therefore, the knowledge could be easily formalized and hardwired into the corresponding knowledge sources. However, most of sub-domains of the damage control domain are either too

complicated or ill defined to be hard-coded. In this case, the corresponding knowledge sources are made capable of learning by means of using appropriate learning modules. The learning modules will be capable of both supervised and non-supervised learning on the set of training examples supplied by the simulator as well as taken from reality (e.g., the ex-USS *Shadwell* experiments).

3.5.1.11 Research Challenges

Novelty of the Blackboard Expert system comes from multiple advanced research aspects. These include:

Multiple-knowledge-sources recursive heuristic classification scheduler, which is used to make use of domain-independent concepts like "rule-out hypothesis". Currently state-of-the-art blackboard systems use it for single type knowledge source only.

Decomposition of Supervisory Control task into crisis recognition and causal response subtasks, which allows relatively independent expert systems development as well as intelligent supervisory override and results in a dual-blackboard system.

Flexible control, which allows for decision making explanation and critiquing. The explanation feature is of high importance for the project as it provides human supervisors with confidence in system's actions. The research aspect here is generating human-readable explanations having multiple type knowledge sources (e.g. neural nets and belief networks).

Learning component for blackboard architectures, which consist of sophisticated subroutines on credit assignment and learning routines for different nature knowledge sources. One of the challenges here is *coordination* of knowledge refinement for different knowledge sources.

Large learning component, which assumes utilizing the simulator to train the expert systems on hundreds of thousands of training examples.

Look-ahead simulation, which uses the simulator for predictive validation as a part of early crisis recognition subsystem. The research issue here is an automated and intelligent training samples development.

Intelligent override component, which will have a sophisticated override component allowing for *smart* interruption by supervisors. Integrating this feature into the blackboard architecture is a research area.

The machine-learning module provides a means of training the system to recognize crisis situations. The ship simulation subsystem provides hundreds of thousands of positive and negative examples of crises. Neural network and symbolic learning techniques will be used to recognize the patterns of features associated with crises. The process provides a way of improving discrimination, so as to minimize false positive and false negative examples. Each method of knowledge-representation—Bayesian and neural networks—has a unique training paradigm.

3.5.2 Neural Networks

The primary advantage of artificial neural networks (ANNs) is that they do not require a human to specify the domain theory, unlike rule-based and most symbolic methods [Fu,

1994]. In the damage control domain, humans have yet to gain enough experience to devise a complete or accurate domain theory. Since the Supervisory Control System project has many novel aspects, this capability is essential. The ANN will devise its own internal representation of the problem when presented with a set of pre-classified training examples [Hassoun, 1995]. The rule-based system will rely upon the ANN to discover rules and help develop the domain theory [Fu, 1994]. Figure 38 provides an overview of how ANNs will be used in the project and how they could interact with the rule-based modules.

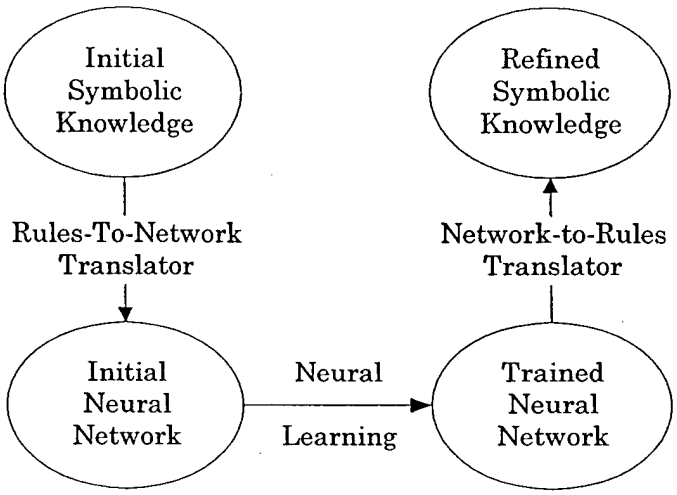


Figure 38. Rule refinement using Neural Networks

In addition, a network with a single hidden layer, or *general classifier*, is able to model any function, provided the ANN possesses enough elements [Hassoun, 1995]. Since the type and scope of knowledge in the Automated Supervisory Control System project is broad and disparate, other learning methods will fail to solve some of the problems. Given enough processing elements, the ANN will find a solution to even the toughest damage control tasks. The chief disadvantage with this capability is that there is no way to determine *a priori* the number of elements required, or the number of training steps necessary, to converge to a solution with a desired degree of accuracy [Hassoun, 1995]. Figure 39 illustrates a typical ANN processing element.

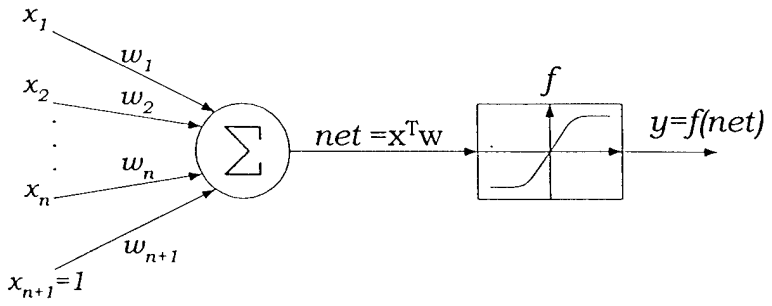


Figure 39. Processing element

Also, an ANN, once trained, executes faster than other systems developed through learning methods. Since the intended situational awareness system is targeted for real-time use aboard naval vessels, fast execution is essential. Systems that propagate knowledge slowly could lead to mission failure.

The basic ANN topology consists of three layers—an input layer, which receives some pattern, a hidden layer that does intermediate processing on the input pattern vector, and an output layer, which does final processing and outputs the classification of the input pattern. Each element in the hidden layer acts as summer, applying a unique function on the input vector, while the output layer treats the hidden layer's outputs as input and in turn applies a unique function on the hidden layer outputs. Figure 40 illustrates one possible way to use an ANN in the damage control domain, in this case, to perform sensor fusion and detect an alarm condition. The inputs to the network consist of multiple sensors -- compartment smoke-detectors, fire-pump sensors, flooding sensors, hull integrity evaluators and environmental monitors -- while the output layer indicates the presence or absence of various alarm conditions, such as fire incipient, flooding incipient, or loss-of-balance incipient.

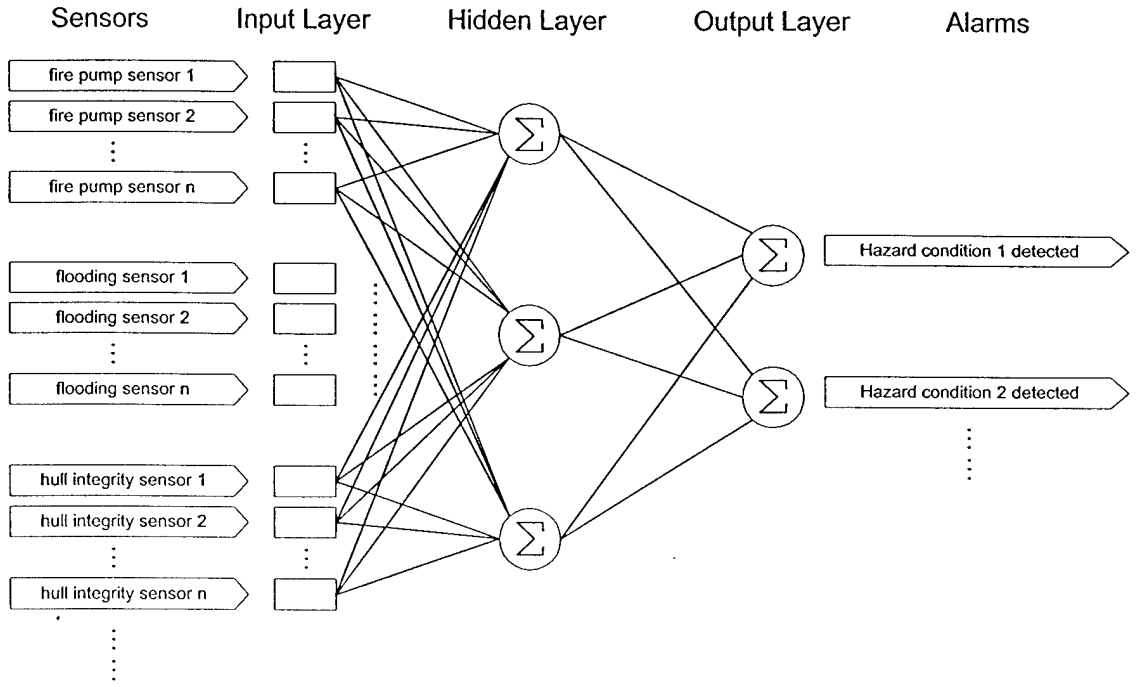


Figure 40. Artificial Neural Network

The most popular ANN training technique is backpropagation [Rumelhart et al, 1986], which uses gradient descent to optimize some error function. Since backpropagation has been avidly investigated by hundreds of researchers, there are various methods of optimizing the algorithm for extremely fast computation. An optimized backpropagation ANN implemented in the Supervisory Control System will permit real-time active-duty modification and improvement of the situational awareness system. With the ANN module, it should be possible for each version of the situational awareness software continuously to customize itself to a given ship on a given mission.

Much of the information for the Automated Supervisory Control System will be encoded in rule tables, and the learning modules must be able to take advantage of them. The advantages of rule-based systems include ease of understanding and need for only a small number of training examples. A hybrid rule-based/ANN approach should be able to combine the advantages of both and yield a superior learning paradigm. Shavlik and Towell [1989] have explored combining a neural network with Explanation Based Learning and have achieved encouraging results. LiMin Fu [1994] has proposed a novel method to extract rules from an ANN once it has been trained. In Figure 41, we see in example "a" the domain theory, which captures some set of knowledge about the damage control domain. Using a strategy like that employed by Shavlik and Towell [1989], we convert the rule-base into a tree, (see Figure 41/b) letting solid lines represent excitatory connections. Finally, we add additional links, letting dashed-lines represent inhibitory connections, to complete the ANN and then perform learning, as in example "c". This process was originally illustrated in the first two bubbles of Figure 38.

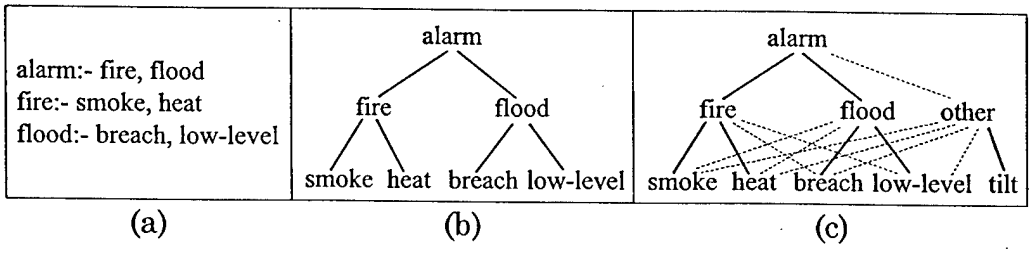


Figure 41. Translation of domain knowledge into ANN

ANNs have been successfully applied to a wide range of learning applications. For example, the post office uses an ANN system to perform handwriting recognition [LeCun et al., 1989] and others have developed ALVINN, Autonomous Land Vehicle in A Neural Net, to drive a van on interstate highways [Pomerleau, 1991].

3.5.3 Belief Networks

Belief networks are currently the most studied method of reasoning for domains that involve uncertainty. There are several advantages associated with belief networks. First, unlike neural networks and rule-based reasoning, they have a very strong theoretical foundation for reasoning under uncertainty, namely probability theory. Second, they provide a framework that allows for integration of associational knowledge with causal knowledge. A ship has a rich amount of causal structure-function knowledge, and intelligent reasoning methods are well served if they are able to incorporate this knowledge into the reasoning process. Third, belief networks can be used for learning as well as for problem solving, and combining these two aspects of expertise are important in the Automated Supervisory Control System development effort.

Belief networks have come to dominate the field of reasoning under uncertainty in artificial intelligence, and new or improved algorithms for belief propagation in such networks are being developed continuously. Of particular interest to the project is the ability of belief

networks to handle sensor fusion in a principled manner. Sensor fusion, including false alarms, is one of the damage assessment problems the system might have to face.

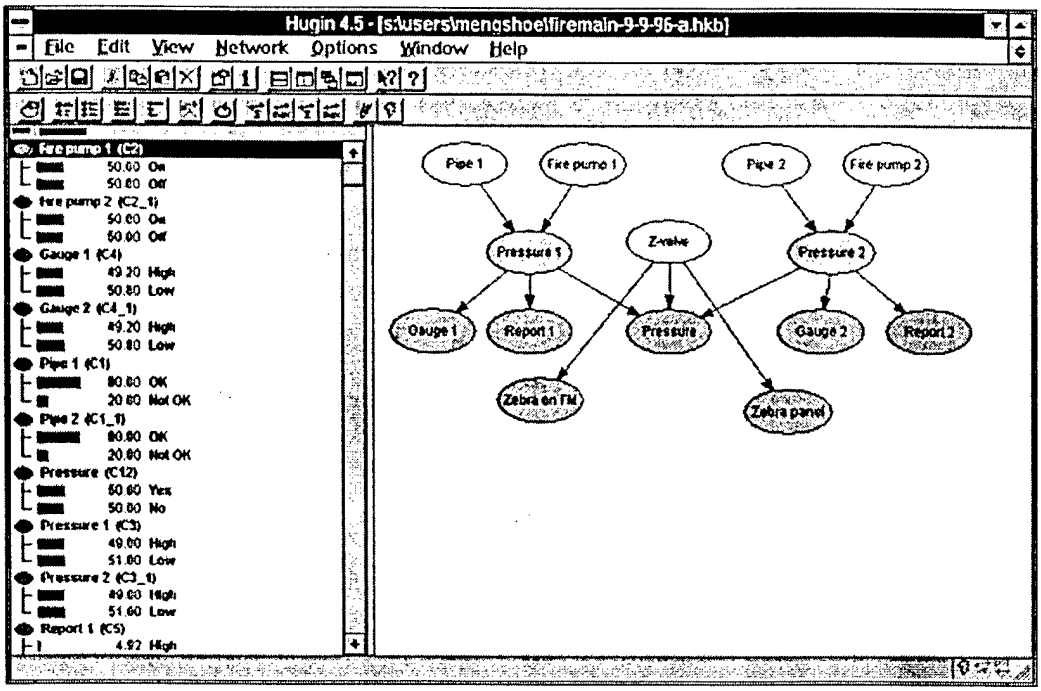


Figure 42. Belief network representing parts of the fire main system

The approach taken to sensor fusion in belief networks is to build a sensor model. A sensor model expresses the dependencies between the variables that are being sensed (representing fires or floods) and the sensors. In particular, the reliability of sensors and the fact that they can fail is expressed in the sensor model; this means that the belief network can make a correct diagnosis (damage assessment) even though some sensors fail or give inaccurate results. An example belief network related to fire main diagnosis and repair is shown in Figure 42. The belief network is shown to the right in the Figure. To the left, the joint probability distribution, calculated based on the belief network and the evidence received so far, is shown. In the network, the nodes Gauge, Report and Zebra panel are sensor nodes, the Pressure nodes are intermediate nodes, while the Pipe and Fire pump nodes are diagnostic nodes. This network allows the situation assessment module to evaluate the probability of rupture on one pipe, say Pipe-1, versus the probability of rupture on another pipe, Pipe-2. Such evaluation would be with respect to the evidence E received; for example one could have E = {Gauge-1 = High, Report-1 = Low, Zebra panel = Open, Gauge-2 = Low}. Now the belief network is used to compare P(Pipe-1 = OK | E) versus P(Pipe-2 = OK | E). This is the probability of Pipe-1 being ruptured (not OK) versus Pipe-2 being ruptured.

In general, the sensor model of a belief network is just the conditional probability table associated with a sensor (evidence) node. The arrow needs to go from the state node to the sensor node. One can have one sensor variable measuring one state variable, or several sensors measuring one state variable. An example of the latter is the following case of sensor (or data) fusion in the context of pressure sensors. Suppose we have two pressure sensors

with error margin +/- 2 PSI, and that one shows 150 PSI, the other 154 PSI. We conclude that the pressure is approximately 152 PSI. Sensor fusion in belief networks can also be used when sensors are very different, as long as the graphical structure of the network as well as probability theory applies. Sensor failure can also be modeled. For any actual pressure (nodes Pressure 1 and Pressure 2), any sensed pressure is allowed. In more elaborate models, the state of the sensor is represented by a separate node.

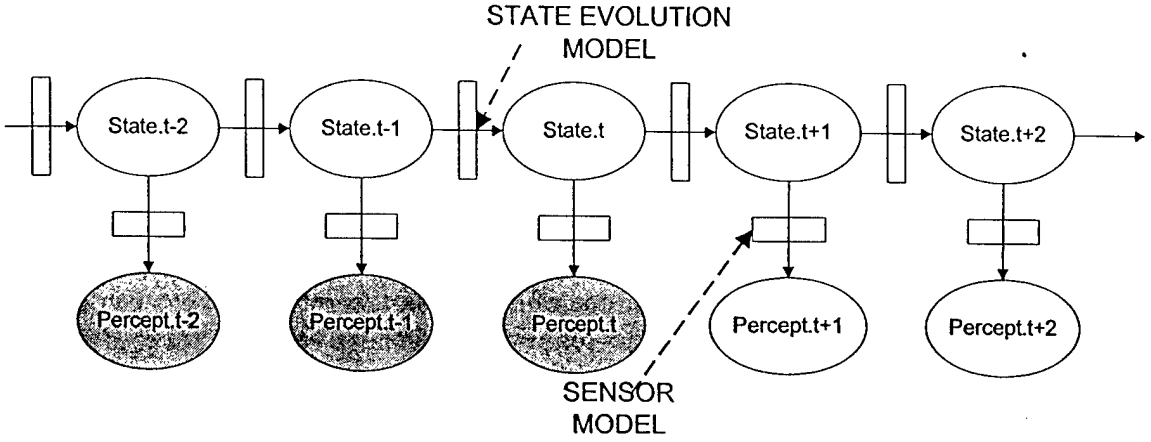


Figure 43. Example dynamic belief network (from [Russell and Norvig, 1995])

Two dimensions that are naturally added to belief networks are those of *decisions* and *values*. When this is done, we get a *decision network* (or influence diagram) [Jensen, 86]. In general, a decision network is used to compute maximum expected utility, and to find the decision that gives maximum expected utility.

Belief and decision networks are static models. Belief networks, for instance, express the joint probability distribution over a set of random variables at a particular point in time. However, change and dynamic behavior are inherent in real-world domains such as the damage control domain. Belief networks can be extended to *dynamic belief networks*, where such issues are addressed. A dynamic belief network is a belief network plus a state evolution (transition) model, where a belief network can be decomposed into a sensor model and a state dependency model. This is illustrated in Figure 43.

Both the state evolution model and the sensor model are based on having a set of random variables and conditional dependencies between them. The state variables describe the current state of the world, while the sensor variables describe the state of the sensors. Notice that one sensor node in the above figure typically represents several nodes in the underlying belief network. The same applies to the state node. Also, notice that a stationary sensor model is assumed:

$$\forall t P(E_t | X_t) = P(E | X).$$

In other words, the behavior of the sensor model does not change. This assumption simplifies the modeling process.

A *dynamic decision network* (DDN) [Basye et al., 1992; Russell and Norvig, 1995] is a dynamic belief network augmented with utility (value) and decision (action) nodes. An example DDN is shown in Figure 44.

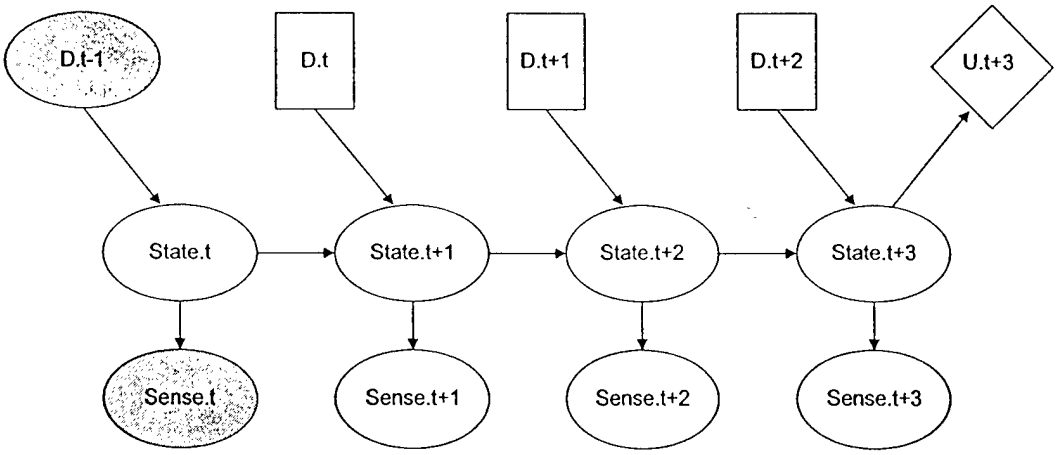


Figure 44. Example dynamic decision network (from [Russell and Norvig, 1995])

Based on a dynamic belief network, the notion of a decision-theoretic agent can be defined [Russell and Norvig, 1995]. A decision-theoretic agent calculates updated probabilities for the current state, calculates outcome probabilities for actions, and selects an action with highest expected utility. A decision theoretic agent utilizes a DDN. Uncertainty propagation in DDN's can be computationally complex. To counteract this, the Markov property is often assumed, making the current state dependent only on the previous state and the previous action. More generally, DDN structure can be exploited. For decision-theoretic planning based on Markov decision processes, three classes of structure have been identified: structure in state space and in phase space as well as structure in decision space [Dean, 1994]. Methods of state space structuring mentioned by Dean are factoring, restriction, functional dependencies, abstraction, and variable creation. Structure in decision space applies to discrete-time, sequential decision problems, a model which is compatible with the system's application. The types of structure in decision space identified by Dean are time-separable, state-separable, and action-separable problems. These structure typologies may be exploited by algorithms that automatically perform (parts of) the Supervisory Control task.

3.5.3.1 Belief Network Challenge: Automated framing of decision problems

There are many research opportunities related to belief and decision networks. This subsection and the next two describe three research challenges of particular interest to the Automated Supervisory Control System. In the following, we discuss and give examples of all three in the context of the system.

Framing a decision problem means identifying the relevant distinctions and relationships, and constructing a decision model at the appropriate level of detail [Selman et al. 1996]. Much work needs to be done before we understand how to tractably determine which distinctions and dependencies are relevant in a particular situation. Consider the fire main system of the ship and how it is supposed to be segregated at condition Zebra. This segregation is shown in Figure 45. For certain situations this conceptualization of the fire main is appropriate; for other situations it is not. One issue is therefore to understand and formalize conceptualizations of the fire main system that are flexible. Another issue concerns how belief and decision networks can be constructed automatically, and how they relate to the functional schematics of the fire main as shown in Figure 45. The novel aspect of the solution approach is that the fire main system is a system for liquid flow, while much

previous work on model-based reasoning has concerned electronic circuits. The above two issues, of course, carry over to other systems on the ship as well, such as the chill water system.

As Figure 45 shows, the fire main is quite complicated, and still only a subset of the fire main system is shown. However, several issues are illustrated through this Figure, and they can all be related to the notion of automated framing of decision problems. There are several specific research challenges related to this notion in the context of damage control: domain size, contexts, and design-time construction. We discuss all three issues in the following.

The size of the damage control domain poses a research challenge. Consider the size of the ship itself as well as the number of systems and the complexity of each individual system. An example is the fire main system, of which a subset is shown. If all of these systems were to be modeled at a detailed level in one belief network, 10,000's or 100,000's of nodes would have to be created and connected to form a network. This network would then have to be used for real-time belief propagation, which is beyond the present state of the art. In addition, the construction and maintenance of one monolithic decision network of that size is problematic. Dynamically constructing a smaller decision network, relevant to the situation at hand, would address all these problems. How to do this is the research challenge.

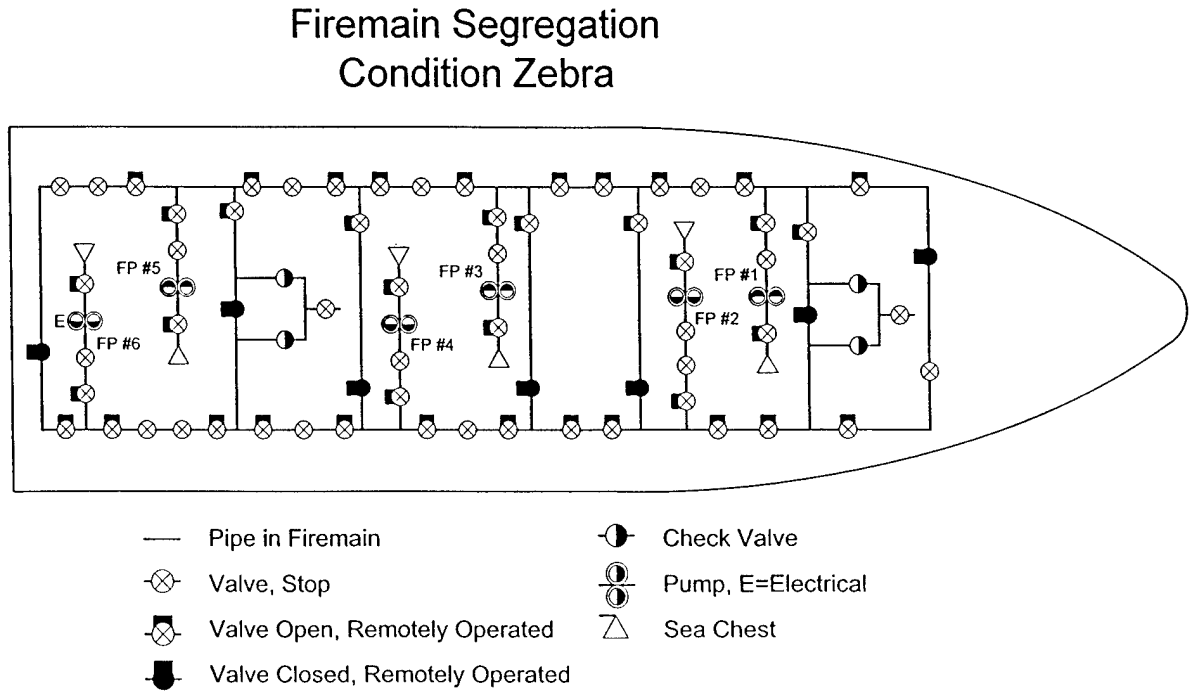


Figure 45. Fire main segregation

Regarding contexts, there are many factors that influence the framing of a decision problem. For the Supervisory Control System, examples of such factors are: materiel condition (Zebra, Yoke), wartime/peacetime, mission, and damage. An example of how materiel condition Zebra affects the preferred status of the fire main system is shown in Figure 45. Now consider how a decision network along the lines suggested by the small examples above may be constructed. For each context, a different such decision network will have to be created.

Furthermore, there will probably be a very large number of contexts. For example, it is quite easy to come up with examples in which the fire main segregation suggested above should be violated, even though Zebra has been set. It might be that such violations mean that a different decision network needs to be created as well. Automated framing of belief and decision networks seems to be a suitable approach to address these problems. The research challenge is to understand and formalize in the language of belief and decision networks how contexts affect such automated construction.

With design-time construction we mean the following. One could use decision networks to search for better (ship) designs. When designing an artifact, its use should be taken into account. A decision network shows how a given design can be used. In particular, the decision networks developed for the project will be concerned with the use of the ship from a damage control and safety perspective. We suggest that these decision networks could be utilized during ship design. For example, one could construct different "what if"-scenarios regarding the type and placement of sensors, to find out the impact of such placement on the overall decision making quality during a crisis situation. In general, the research challenge addressed is to consider the reliability and safety implications of various ships designs as early as possible in a ship's life cycle.

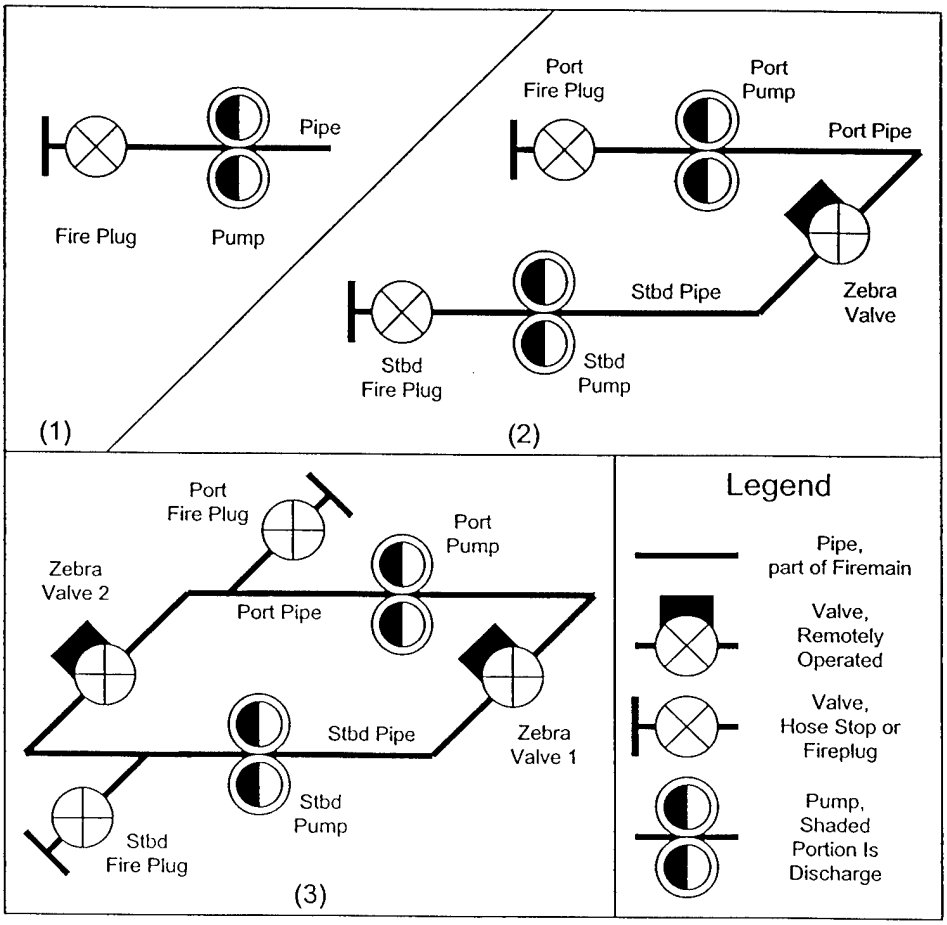


Figure 46. Parts of the fire main system

The research direction can be illustrated as follows. Consider small fragments of a hypothetical fire main system, for example as illustrated in Figure 46. Now consider how one might represent these fire main fragments in the form of a belief network. First, consider the simplest fragment shown in Figure 46, fragment (1). How this can be represented as a belief network is displayed in Figure 47. Figure 48 shows how the fire main fragment can be represented as a belief network.

The above two examples focus on the relationship between a ship's structure and belief networks representing that structure. There are other sources of knowledge that can be utilized. First, the current situation needs to be taken into account. That is, a constructed decision network will have to be constructed based on the situation at hand. Second, a damage control supervisor will have certain responsibilities and goals, and those can also be utilized as well. Third, the communications to and from the supervisor can be utilized. One possibility is to parse these speech acts, and to construct belief and decision networks during this parsing process. This would be similar to compilation or natural language parsing.

3.5.3.2 Belief Network Challenge: Handling time, synchronicity, and streams of events

Handling time, synchronicity, and streams of events is a research challenge at the core of the Automated Supervisory Control System development effort. Research on acting under uncertainty has generally assumed a temporal world. Systems need to represent and reason about the time-dependent dynamics of belief and action, including persistence and dynamics of world states. Better means of synchronizing an agent's perceptions, inference, and actions with important events in the world also need to be developed. An example of this is causal reasoning about effect of actions, as in: `What are the most likely effects of repairing starboard rupture in place?' Consider the belief network showing how parts of the fire main system can be represented. Suppose Starboard Rupture has been established, i.e. `Pipe 1 = Not OK.' Now the DCA needs to evaluate the decision `Repair Starboard Rupture in Place.' This can be done by temporal forward projection over the belief network, resulting in an increase in the probability of `Pipe 1 = OK.' The utility of `Pipe 1 = OK' is higher than `Pipe 1 = not OK,' and therefore the action is recommended. The research challenge is to model the many processes taking place, and how they evolve over time and need to be synchronized.

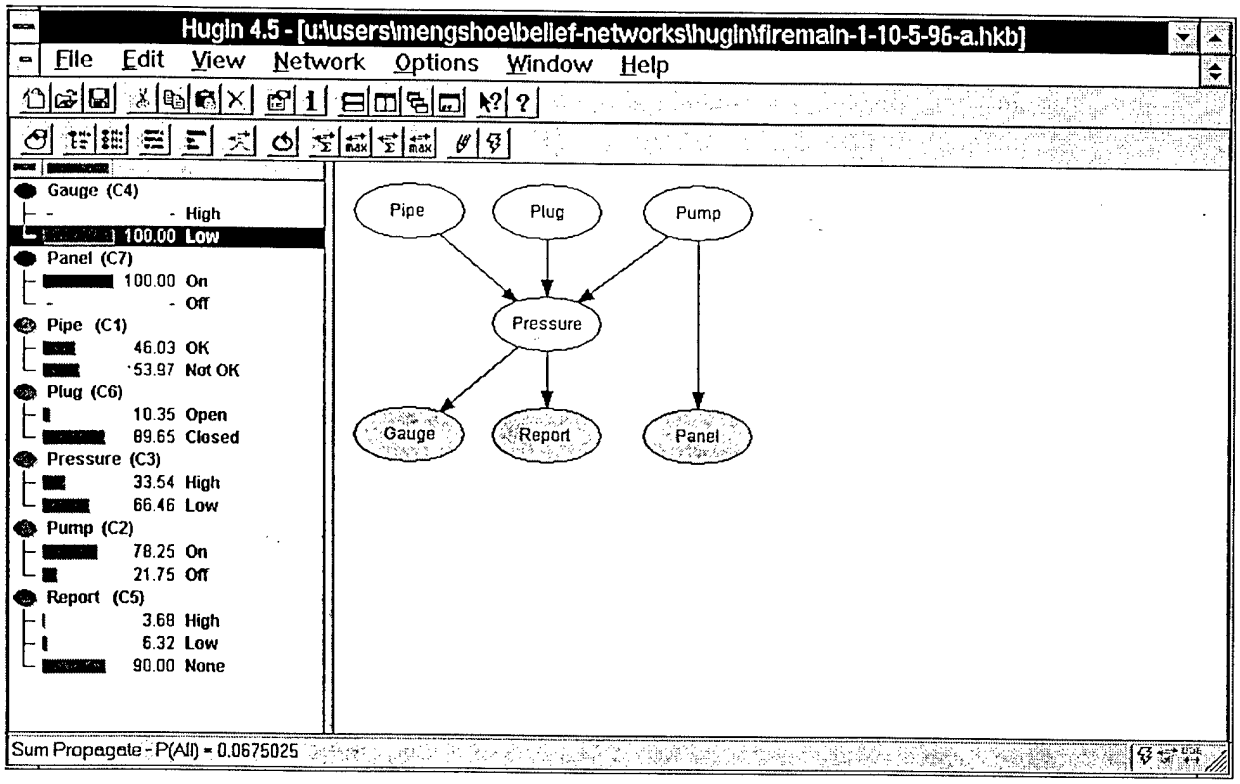


Figure 47. Belief network representation of part of fire main system

3.5.3.3 Belief Network Challenge: Modeling Preferences and Utility

Modeling preferences and utility is important in many domains, particularly in this one. Utility theory provides the principle of maximum expected utility: an agent should take actions that maximize its expected (or average) utility (or reward). Where does information about utility of states come from? Whose utility is being maximized? How can we derive utilities associated with solving sub problems from assertions about high-level goals or from utilities on goal states? What is the most reasonable utility model for evaluating finite sequence actions an agent might take over time? These are all important questions that we expect to address as part of this field of research.

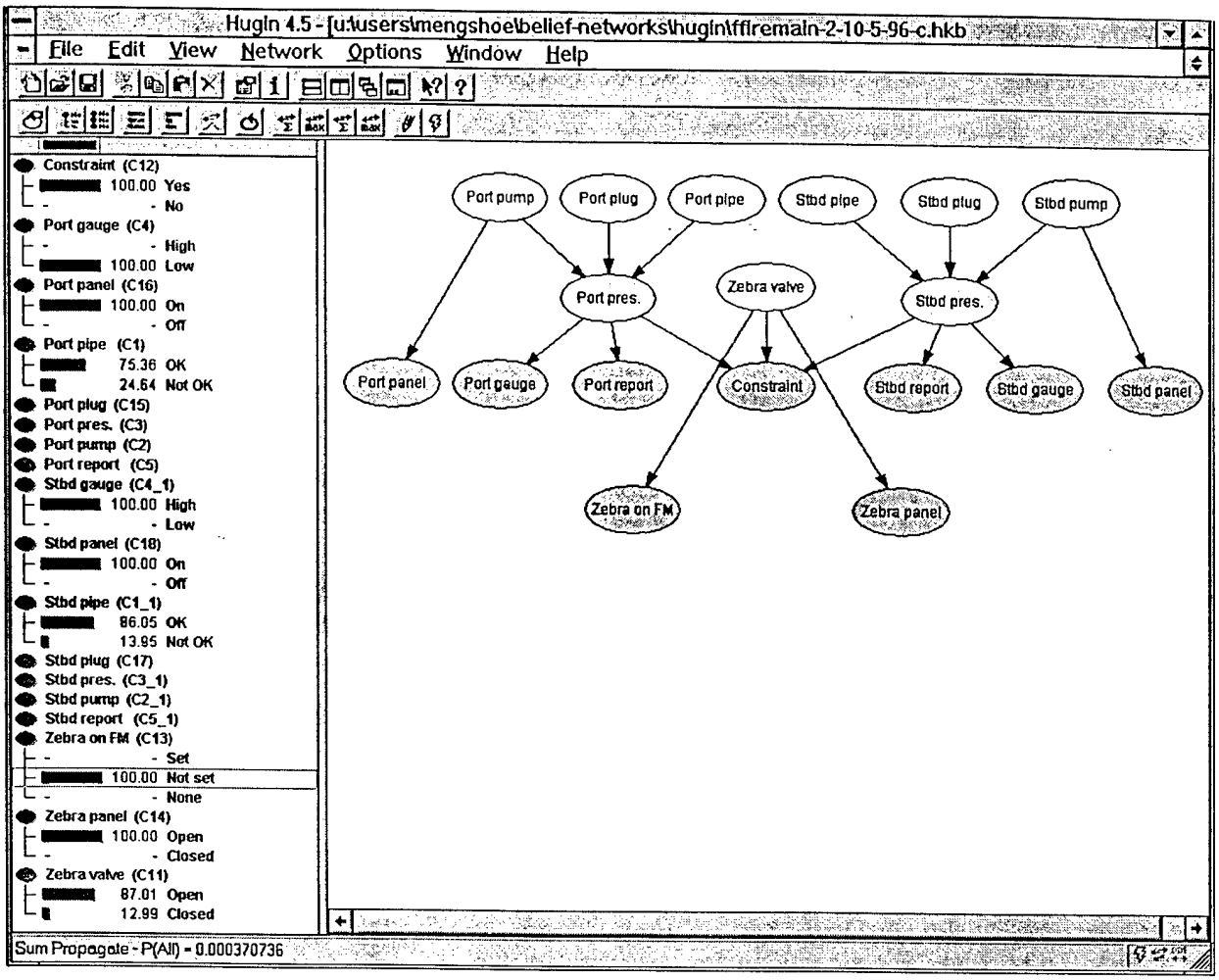


Figure 48. Belief network representing larger part of fire main system

Reasoning about the trade-offs between different courses of action is possible in a decision network. An example of this is: `Should the fire main be isolated before repair or should the fire main be repaired in place?' (Isolation means that fire main valves are closed, meaning no water is supplied to parts of the fire main for some time. This is an obvious disadvantage if there is a fire close to those parts of the fire main. Repair in place, on the other hand, gives some water all of the time. Isolation gives faster repair than repair in place, however.) This problem might be addressed in a DDN by a combination of temporal forward projection, decision nodes and value nodes. One can envision two decision nodes `Isolate Fire Main before Repair' and `Repair Fire Main in Place,' and value nodes that reflect the value of having a quick repair and no water for some time versus a slow repair and some water all of the time. This should give different values for the two decisions depending on the circumstances, as we would like. Exactly how to go about encoding the above behavior in a decision network is not clear, however. This is the research challenge.

A related example of how the DCA thinks about the effect of his actions on the future state of the ship is the following. Within an extended, classical planning framework, the following behavior would be called construction of contingency plans. Suppose that the sprinklers are activated in a ship's magazine room. In this case, the DCA may ask the commanding officer for permission to flood the magazine *before* knowing whether the sprinklers fail or not. The

point here is that the probability of such sprinkler failure is quite low; however, the associated utility is extremely high. That is, if the sprinklers do fail, there is a good chance that the magazine room can reach a critical state unless immediate action is taken. The only action available is to flood the magazine; however, performing this needs to be approved by the captain, and such approval takes time. Thus, the DCA may prefer to get approval in advance, in order not to lose time if it gets necessary to flood the magazine. At an intuitive level, the above line of reasoning seems reasonable. However, even though the framework of belief and decision networks seems suited to support this type of reasoning, it is a research issue exactly how this should be done.

3.5.4 Crisis Predictive Validation Module

Under the rigorous conditions to which a ship at sea is subjected, even the most reliable sensors often fail. Because all of the future ship's damage control systems will depend on sensor readings, it will be necessary to ensure that these readings are accurate. A low (hardware) level of error detection and error correction will be implemented within the smart sensors themselves. However, the critical nature of the damage control task requires an even higher degree of reliability than can be provided by hardware-based systems alone. A higher (software) level of sensor response verification is necessary.

A process known as Predictive Validation can be used to facilitate this capability. It combines an accurate physical simulation of the ship with an intelligent reasoning system to perform verification of sensor indications. When a crisis is suspected, the outputs of sensors pointing to the crisis are input into the ship simulation subsystem, which is able to simulate the development of the incipient crisis much more quickly than it would evolve in real time. The intelligent reasoning system working in conjunction with the simulation module then determines what response is expected from the sensors a short time into the future based upon their initial indications, and the expected development of the crisis. If the actual sensor response matches the simulated response, there is a high probability that the crisis is real. If, however, the latest sensor readings do not match the predicted response, one of two possible scenarios is true. Either the sensors are giving erroneous readings and the crisis is not real, or the rule base used by the intelligent reasoning system to model the evolving crisis is faulty. In this case the output of the Predictive Validation module is not conclusive, but at least it alerts the human Damage Control Assistant to the necessity of having the area of the ship in question investigated by other means.

Thus, the main advantage of the Crisis Predictive Validation module is the added degree of confidence in the ship damage control systems that it provides the situational awareness expert system. Whenever a match is indicated between predicted and actual sensor reading (which should be the vast majority of the time), the SA expert system can assume that the readings are correct and can confidently take immediate action to stem the nascent crisis without the need for further verification. If a divergence between actual and predicted readings is found, the suspected crisis can be further investigated. The key point, however, is that this divergence should occur in rare cases, thus the large number of personnel normally involved in investigating activities can be reduced to a minimum.

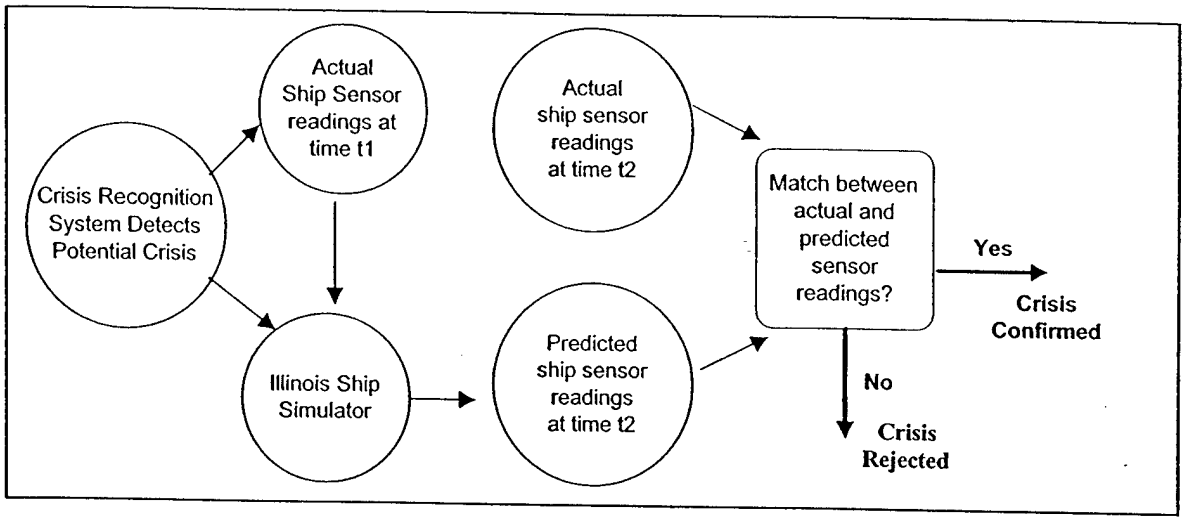


Figure 49. Crisis Predictive Validation Module interaction

3.5.5 Intelligent Ship Objects

The intelligent objects module provides an interface between the smart sensors and the Supervisory Control system. Initially, the behavior of smart sensors will be simulated by the intelligent agents module: they will react to a spreading fire appropriately. As the smart sensors are actually implemented on the ex-USS *Shadwell*, the module will serve to interface the smart sensors to the Supervisory Control system.

The intelligent objects module for smart actuators provides an interface between the smart actuators (water mist, sprinklers, cut-off valves) and the Supervisory Control system. Initially, the behavior of smart actuators will be simulated by the smart actuator module, thereby changing the course of a spreading simulated crisis. As the smart actuators are actually implemented on the ex-USS *Shadwell*, the module will serve as the interface of the smart actuators to the Supervisory Control system.

The intelligent objects module for ship components simulates their behavior during a crisis, such as the state of the air conditioner units, chill water system, fire pumps, fire main, and electrical system. The ship components module provides a means of understanding component deactivation during a crisis.

3.6 Supervisor Performance

A major determination of the success of the Automated Supervisory Control System relates to the performance of the system supervisor. Although the number of damage control personnel will be reduced with the system, and the system is intended to require only minimal supervision, the person charged with supervising the system will be more significant to damage control than any individual engaged in current damage control operations. The system supervisor will have enormous responsibility, making the decision about whether or not to override the system and transfer damage control activities from the system to the hands of humans. Disaster may result from an incorrect decision to override the system, as well as to not override the system. Thus, effective supervisor performance is critical to the success of the Supervisory Control System. Here we address the major issues involved in human performance in conjunction with the automated system.

One key issue is whether the interface gives damage control supervisors confidence in their acquiring a sufficient understanding of an evolving crisis, since such information serves as the basis of how the damage control expert would override the system. That understanding is analogous to the situation assessment acquired by the automated system, and is discussed in section 3.6.2 as "Supervisor Situation Awareness". Another key issue is whether the supervisory interface adequately supports the supervisor in decisions to *manually override* the system. This is discussed in Section 3.6.3. The third key issue is whether those responsible for damage control on a ship have sufficient *trust* in the supervisory interface to enable them to coordinate their actions with it. Because this issue relates closely to problems of system evaluation, it is covered in section 3.7.

In several respects, research on human interaction with an automated system is similar to that on human interactions in teams. It is worth noting that research in the latter area has benefited from an emphasis on cognitive processes [e.g., Sniezek, 1990, 1991, 1992; Sniezek and Buckley, 1995; Sniezek and Henry, 1989, 1990; Sniezek, May, and Sawyer, 1990], and on decision-making under stress [Salas, Bowers, and Cannon-Bowers, 1995]. The solution approach is intended to yield a better understanding of the cognitive processes of a supervisor of a complex automated damage control system through empirical tests of Supervisor Situation Awareness (Section 3.6.2) and Supervisory Control Decisions (Section 3.6.3). Also, the research is designed to incorporate the influences of acute stress on decision making, with particular attention to time pressure, increased cognitive load and anxiety (Section 3.6.4). Much of the testing on individual components of the interface can be done with student participants in laboratories at the University of Illinois at Urbana-Champaign, as indicated in Section 3.6.5. Additional refinement of the supervisor interface will require input from damage control experts.

Supervisor performance is best understood in the context of interactions with the automated system. Thus, we comment on issues pertaining to the supervisory interface prior to discussing supervisor situation awareness and control decision making.

3.6.1 Interface Considerations

A significant and ongoing part of the project is preparation for coordination between the automated system and personnel on ship via the supervisory interface. The complete design plan for the supervisory interface is presented in detail in Section 3.2. Here we are concerned

with tests with human subjects of those components that allow communication between a system supervisor and the automated system (the graphic visualization and control panels). It is important that end-users and human factors specialists should be involved in early development [Madni, 1988]; results of tests during development can be used to improve the interface design. In the initial phase of the work, efforts will concentrate on the development of the supervisory interface that supports the supervisor's understanding of, and response to, communications with the system. We intend to establish through subsequent testing that the final interface meets requirements.

The supervisor interface is a crucial link between the automated system and personnel aboard a ship. For successful coordination, the interface must allow for Supervisor Situation Awareness (which is elaborated upon below). That is, the interface must communicate sufficient information that the supervisor can achieve and maintain the necessary level of situation awareness in real time. We propose testing throughout development of the interface to allow for ongoing modification to the interface design for the purpose of achieving this requirement. The psychology literature identifies numerous potential influences on the extent to which Supervisor Situation Awareness can be attained. These lead to questions such as: How much activity is needed on the part of the supervisor to maintain the vigilance needed to achieve situation awareness and detect changes in the state of the ship? How can supervisors discriminate among changes indicating events needing no intervention, events needing monitoring, and events needing action? How much information should be provided automatically to the supervisor vs. available only after a query by the supervisor? How can information be managed so that cognitive overload is prevented during conditions producing acute stress?

Recommendations for the design of the interface can be obtained from past research. However, experimentation will be used to learn how to adapt features of effective supervisory interfaces to the problem of damage control on a ship.

3.6.2 Supervisor Situation Awareness

It is not sufficient that the system achieve automated situation awareness; the supervisor must do so as well. Supervisor Situation Awareness refers to the supervisors' mental model of critical events and understanding of their interrelationships at the time they are occurring. The supervisors' situation awareness is appropriate if the supervisors can detect critical events and make inferences about the outcomes of these events; only then can the supervisors select correct actions in response to varying states.

Given the size and complexity of a ship, situation awareness in the absence of automation is not possible without highly coordinated communications among numerous personnel. Even then, it cannot be achieved in a timely fashion. With an automated damage control system, situation awareness will be difficult, but not impossible, to achieve within a short duration of time. The sensors monitoring the condition of the ship will be providing large amounts of data that are not only complex, but also changing rapidly. The system operator will need to attend to, interpret, update, and make inferences from these data on a continuous basis, especially during an emergency. For situation awareness to be complete, the operator will have to make inferences beyond ship data factually presented by the damage control system [Sheridan, 1992]. In addition, the operator must be prepared to intervene, make rapid decisions, plan and execute actions in the event of system failure. These actions are not

possible if the supervisor does not have adequate knowledge about the functioning of the automated system.

Consider the case of one China Airlines 747, which had an engine failure while the plane was on autopilot. With the loss of an engine, the plane tended to bank to the right. The autopilot fought this motion in order to maintain the plotted course. The captain, not realizing the cause of this action by the autopilot, turned it off, causing the plane to lurch to the right, which produced a crisis [Adams, Tenney, and Pew, 1995]. A lack of situation awareness by the operator can lead to unrecoverable damage. Even with extensive automated situation awareness, the operator will need awareness of the ship's condition to monitor and respond as necessary.

Thus, it is necessary that research on the supervisory interface assess the situation awareness of the supervisor. The proposed work includes tests aimed at capturing supervisor situation awareness, and modifying the interface as needed to enhance it. The goal is to verify the adequacy of the supervisor's situation awareness under a variety of simulated crisis conditions. Situational awareness will be measured in tests such as the one outlined late in Section 3.6.5.

Techniques for measurement of supervisor situational awareness will be based on those discussed in the literature on situational awareness. One feasible technique for measuring situational awareness is a freeze-frame questionnaire used by Endsley [1995]. This procedure consists of freezing the task (and blanking the screen/displays) at a randomly determined point in the task. Although it is not necessary to continue the task after the freeze-frame, doing so does not impair task performance. Thus, it has been concluded that the freeze-frame technique is only mildly intrusive and does not suffer from memory decay (at least over 5-6 minutes) [Endsley, 1995].

Other potential procedures for measuring supervisor situation awareness include the use of "external task measures." These require changing external information, removing displays, during the course of the trial and seeing how long it takes the participant to react. Another category is that of "embedded task measures." These demand an action on the part of the supervisor that can be completed correctly only if situational awareness is present. For each of the above techniques, it is possible to apply policy-capturing techniques to describe the person's strategy for using multiple sources of information to make judgments about the situation [e.g., Sniezek, 1986, 1988]. Finally, it is possible to query supervisors directly via an interview or checklist about their recall of information presented via the interface [Endsley, 1995].

3.6.3 Supervisory Control Decisions

Theory and models of supervisory control show the complexity of the problem of removing human behavior from the loop, while still maintaining human potential for intervention in the system. Past research with automated systems (such as aviation automation, nuclear power plants, and supervisory monitoring displays) identifies key problems for the supervisor if the system fails, including failure detection, diagnosis, and location. In event of a system failure, there will be many threats to supervisor performance of these tasks [Wickens and Kessel, 1980]. And even if the supervisor manages these tasks well, there remains the significant problem of effecting damage control manually. In short, the decision to take control from the automated system will be the most difficult and demanding task facing the supervisor.

Problems can be expected from the number and nature of demands on the supervisor. When there are several distinct events to monitor and one or two require extra attention, other important events occurring elsewhere may be overlooked. This appears to be what occurred after an air traffic controller at Los Angeles International Airport cleared a commuter plane to hold at a takeoff position. The controller was then distracted. When her attention returned she noticed another commuter plane near that runway and directed it to stay off the runway. This second commuter plane was mistaken for the first. A landing jet then collided with the commuter plane on the runway [Adams, Tenney, and Pew, 1995]. Investigating interruption events during supervisory control Gillie and Broadbent [1989] found that the biggest problem in returning to the interrupted task was complexity of the interrupting event and its similarity to the original event.

It is clear that successful supervisory control performance requires high levels of attention and perception. But these skills are far from adequate. In order for the supervisor to act on the perceived events, he or she must be capable of making rapid judgments and decisions. Generally speaking, the decisions fall into four classes:

1. Non-intervention, the decision to allow full automation to continue. Note that this results from a choice process, and thus is not the same as the absence of a decision about intervention.
2. Information search, the decision to seek additional information about ship status, advice from the system, or explanation of an action by the system.
3. Low-level system override, a decision to alter sensor data or situation awareness conclusions pertaining to crisis recognition or predictive validation.
4. High-level system override, the decision to take control functions away from the automated system in whole or part, and dictate casualty responses manually.

All of the supervisor's decisions regarding control of the automated system are important to understand. The second class of information search is especially useful for training purposes. Nevertheless, we shall give most attention to the decision with the greatest potential for catastrophic consequences, the decision of system override.

Subordinates will not necessarily override superiors (e.g. "mutiny"), even when the superior is obviously wrong and failure to override means death for the subordinate [Foushee, 1984]. For example, Ruffel-Smith [1979] reports that copilots rarely made the decision to override their captains in a simulation, even when the pilot was incapacitated. The literature on decision-making identifies several factors that can influence supervisory control decisions. From the work of Lee and Moray [1994], we can expect that the choice of manual over automated control will depend on risk, responsibility, Navy policy, mental workload, and trust in the system. Laboratory simulations focusing on the override decision-making process (as suggested in Section 2.6.5) can test the effects of such factors. This part of the research will produce insights that can be applied to training for supervisory control. Techniques to structure the problem and guide the processes of evaluating risks from alternative actions can be beneficial in training and in practice [Sniezek, Kuhn, Spurlock, 1995]. But because supervisory control decisions will ultimately take place in a crisis, the natural context of these decisions is an important consideration. We proceed to survey the anticipated effects of crisis conditions on supervisory decision-making.

3.6.4 Acute stress

Undoubtedly the most significant difference between the crisis conditions under which supervisory control decisions are made and the laboratory conditions under which preliminary testing will take place is the potential for acute stress in the former. As defined by Driskell and Salas [1996], acute stress occurs in conditions of potential harm, time pressure, and arousal. While highly experienced persons performing routinized tasks may not suffer substantial losses of ability during acute stress [Klein, 1996], performance decrements are a more common result [Orasanu & Backer, 1996].

Unfortunately, very little is known about the effects of acute stress on the types of cognitive skills needed for supervisory control. However, research on the effects of arousal on allocation of attention does suggest an effect of acute stress on one of those cognitive processes—supervisor situation awareness. The literature suggests that arousal reduces the amount of information individuals attend to in their environment [Bacon, 1974; Easterbrook, 1959]. Therefore, the supervisor's situation awareness may be reduced under acute stress. It has been suggested that individuals experiencing arousal attend only to the information they perceive as most important [Bacon, 1974; Furst and Tenenbaum, 1985]. It is therefore all the more important to develop a supervisory interface that maximizes supervisor situation awareness and to train supervisor candidates in which information to which to attend in crisis situations.

The impact of acute stress on supervisory control performance is difficult to test experimentally for many reasons [Driskell and Salas, 1996]. Yet it is possible to conduct tests of judgment and decision making performance under the realistic crisis conditions at the ex-USS *Shadwell*. Several questions are pertinent, e.g. what is the impact of a sudden emergency event, such as a missile or mine hit, on the ability of the supervisor to maintain situation awareness and make appropriate supervisory control decisions? Is there initial panic or acute anxiety that reduces the cognitive resources allocated to the tasks of supervisory control?

The delegation of supervisory functions to one or two persons may diminish some benefits currently experienced by the use of teams for damage control. The role of team leader changes dramatically from non-stressful to stressful situations with a decrease in perceived competence and influence [Hamblim, 1958]. The leader is seen as more prominent during crisis situations vs. non-crisis [Isenberg, 1981], with team members significantly more likely to defer to decisions made by the team leader under stressful situations [Driskell and Salas, 1991]. Hence, it may be important to define the role of the supervisor more as the primary decision-maker responsible for damage control than as a servant of the expert system.

It is reasonable to expect individual differences in the tradeoff between data collection and the taking of action by operators. Some research suggests that some automation supervisors are more complacent than others are. Thus, it will be important to set clear guidelines for supervisory control actions. It may also be wise to select for desirable tendencies as well as low levels of trait anxiety. Another option may be to introduce training procedures, such as increased levels of positive self-evaluation [Baumann, Donovan, and Sniezek, 1997] that may reduce the debilitating effects of anxiety on performance.

More is known about one of the characteristics of situations involving acute stress: time pressure. By definition [Maule and Svenson, 1993], time pressure increases cognitive load.

Decision making strategies change under time pressure and decisions appear to be based on less information [Edland and Svenson, 1993].

Another effect of time pressure is anxiety [Leon and Revelle, 1985], which generally hurts performance [Burton, 1988; Hardy and Parfitt, 1991]. Prior work by Baumann, Sniezek, Donovan, and Wilkins [1996] with naval officers in training to be damage control assistants supports this conclusion. This effect may be explained by a decrease in cognitive resources applied to the task under anxiety [Backman and Molander, 1991; Baumann, Donovan, and Sniezek, 1997; Molander and Backman, 1994; Morris, Davis, and Hutchings, 1981]. This suggests that an expert system would aid an individual in a crisis situation to the extent that it reduces the cognitive demands of the task.

3.6.5 Laboratory Studies (Due to the DC-ARM schedule and reduced funding, this portion cannot be completed).

Preliminary experiments on supervisor situation awareness can be conducted in the laboratories at the University of Illinois. Participants familiar with human computer interfaces will be recruited through campus advertisements. Populations targeted will include aviation students and members of ROTC. Participants will be recruited for multiple trials and be financially compensated for their time.

Early trials will include familiarizing the participants with the task and measuring the effect of various display configurations on situation awareness, and perceived expertise and confidence [see Trafimow and Sniezek, 1994]. The next set of trials will involve training the participants to understand the decision rules used by the expert system. This is in preparation for the third set of trials, which will explore when and why participants choose to intervene. Intervention decisions would include querying the expert system for more information, explanations of the expert system's actions, and partial or complete override of the expert system. In the final set of trials, the effects of factors such as time pressure, anxiety, and arousal on decision-making and situation awareness will be explored. Information obtained on the effects of time pressure on supervisor situation awareness and supervisory control decisions will be examined with an eye towards developing recommendations for training and for interface design.

The display configurations used in the initial trials will be determined through consultation with damage control experts. The two display configurations that result in the highest situation awareness amongst participants in the University of Illinois sample will then be compared using damage control experts as participants.

As mentioned in Section 3.6.2, there is an ongoing debate concerning the best way to measure situation awareness. Alternatives we are considering include a "freeze-frame" method (pausing the scenario and blanking out the screen), post-trial questionnaires, and embedded task measures [Endsley, 1995]. For both the freeze-frame and post trial questionnaires, participants will be asked about a variety of situation parameters. For instance, the participants may be asked about fire main pressure readings, the status of the chill water loop, status of various shipboard systems (such as radar), and the tactical situation at the end of the scenario or the time of the freeze-frame. Which parameters are most important for the participants to be aware of will be determined through consultation with damage control experts.

3.7 Evaluation and Testing (Due to the DC-ARM schedule and reduced funding, this portion of the task could not be completed).

Automation does not necessarily improve human performance; hence, it is necessary to evaluate systematically the impact of automation of damage control functions on the behavior of personnel interacting with it. Research is needed to understand what is necessary for personnel to perform effectively with automated systems--particularly as military systems become more automated [Thornton et al, 1992]. Evaluation and testing will be an ongoing part of the solution approach, as well as a significant activity at the conclusion of the project. Testing during system development will allow feedback to the design process. In addition, this testing will produce evaluation data that can be a useful standard for comparison and interpretation of the end results.

In this section we do not reiterate descriptions of those tests that will be conducted for the primary purpose of making design choices and modifications. Rather we describe the general plan for collection of data that will provide final evidence of system and supervisor performance. Because evaluation data will be collected during intermediate tests, these are discussed as well.

There are two perspectives from which to evaluate an automated system: *actual* system performance, and *perceived* system performance. We begin with the latter, addressing in depth that part of subjective evaluations of automated systems known as trust.

3.7.1 Trust

One risk associated with the Automated Situation Awareness System concerns the acceptance of a fully automated damage control system in the US Navy. For the system to be implemented successfully, it is necessary that both users (supervisors/operators) and US Navy decision-makers have *trust* in the system. In general, trust refers to the belief that the agent is reliable and competent, and serves one's own best interests. There are three goals to our research efforts pertaining to the beliefs underlying trust in the system:

1. The first goal is to effect user acceptance of this technology. The literature on trust provides numerous recommendations for influencing beliefs about the system on the part of the user. Many of these recommendations will be applied in the course of the solution approach.
2. The second goal is to obtain data demonstrating user trust in the system. As outlined in section 3.7.2, the solution approach will measure supervisor trust in the system at various stages of development.
3. The final goal is to report detailed information about the success of the system. This includes the results of evaluations described in 3.7.2, and the trust data showing perceptions of success by Navy personnel who have performed with it. These summaries should be valuable to those who will ultimately decide the fate of the damage control technology developed under the Automated Situation Awareness System.

We proceed to review the problem of user trust of technology, and list strategies for achieving the first goal, developing supervisor trust in the system. Figure 50 shows some of the influences on supervisor trust that will be manipulated, and how the resulting trust affects

supervisory control decisions. Then we present a plan for step 2, the measurement of user trust. The third step of the effort on trust will be accomplished through reports on the research, and thus is not discussed further here.

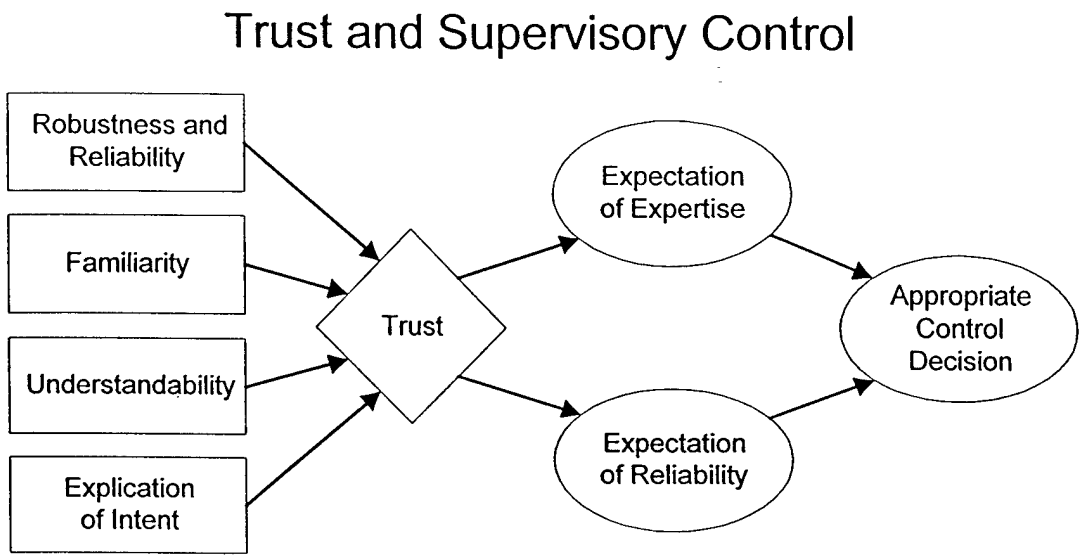


Figure 50. Factors influencing supervisor trust in the system, and the role of that trust in supervisory control decisions

3.7.1.1 Developing User Trust

Sheridan [1992] lists seven causes of trust that can be used to increase trust in the reliability and competence of an expert system: reliability, robustness, familiarity, understandability, explication of intention, usefulness, and dependence. There are several ways in which the proposed work can encourage trust in the system at the level of the user.

First, familiarity can be built into the interface by using representations of the ship, sensor functions, and actuator functions that fit the schema held by users. Exposure to the system during training is expected to increase familiarity as well.

Second, robustness and reliability can lead to trust [Earle and Cvetkovich, 1995]. Robustness refers to the ability of the system to perform in a variety of different circumstances. To the extent that the supervisor can observe and work with the system in a variety of scenarios, trust should increase. Also, to the extent that the system gives consistent advice in similar scenarios, it should be perceived as reliable with a resulting increase in trust. Therefore, repeated exposure in similar situations and exposure to novel scenarios with the expert system are expected to build trust. The ability of the situational awareness system to specify a large variety of scenarios will make it possible for supervisors to acquire the necessary exposure during training.

Third, user acceptance can be enhanced if the supervisor can understand the Automated Situation Awareness System. Not only do users prefer explanations and recommendations they can understand, but these explanations also help the supervisor form a mental model of how the system works. Research has found that acceptance increases if the system explains

its advice [Ye and Johnson, 1995]. Another benefit of making the advice understandable is that the supervisor can determine its correctness [Hollnagel, 1987]. Because the expert system to be created will transfer situation awareness to the operator, and provide a degree of transparency about its reasoning processes, we anticipate that satisfactory acceptance can be obtained.

A fourth approach to building supervisor trust in the system is to implement explication of intention. With other people we tend to trust those who publicly announce their intentions. A similar process may occur with an expert system. Besides understanding the current actions of the expert system, it is important that the expert system tells the user what it intends to do. If the system intends to flood certain compartments or close particular valves, it is important that the supervisor is aware of its intentions, preferably at the time the decision is made.

A final problem deserving attention concerns the role of the supervisor with respect to the system. Navy personnel responsible for damage control may not accept a new fully automated system, particularly if it "...appears to usurp their authority or unduly restricts their options" [Madni, 1988]. Also, users may fear that reliance on an expert system may atrophy their skill or reduce their real or perceived contribution to the job [Sheridan, 1992]. It might be helpful in this regard to define the supervisor as a decision maker for non-routine situations, and establish drills involving scenarios that do, and do not require supervisor intervention.

As indicated in Figure 50, the supervisor trust in the system that is necessary for implementation of the Situation Awareness System can be established via the manipulation of various factors. It is important to note that such trust is likely to affect the appropriateness of control decisions [Lee and Moray, 1994]. Thus, a critical feature of the proposed work is that it endeavors to understand how the supervisor can have an *appropriate* level of trust in the system. There is little value in encouraging operators to trust a system that is destined to fail. In general inappropriate trust or overconfidence in the system can result in sub-optimal decisions and accompanying negative consequences [Olson and Sniezek, 1996]. There are various risks of developing overconfidence in one's own choices [Sniezek, Paese, and Switzer, 1990; Sniezek and Buckley, 1991, 1993; Trafimow and Sniezek, 1994], as well as over trust in the system. The mere dependence of the supervisor on the automated system for information can increase trust [Schlenker, Helm, and Tedeschi, 1973; Sheridan, 1992; Van Swol and Sniezek, 1996]. And, it is possible that increased opportunities to understand system reasoning may not improve supervisor performance, but yet enhance the supervisor's perceptions of performance [Kotteman, Davis, and Remus, 1994], thereby increasing trust more than is desirable. Alternatively, problems of over trust or overconfidence may result from a highly reliable system. In a comparison of consistently reliable system operation vs. variable reliability (as in the completeness or quality of sensor data), Parasuraman, Molloy, and Singh [1993] found evidence consistent with overconfidence in the system. Operators failed to monitor the automated system satisfactorily when it consistently provided reliable information. However, there was no decrement in vigilance with variable reliability. This suggests that rate of monitoring drops as degree of reliability increases. Reduced monitoring of reliable systems may be efficient at times, but it could also reduce the extent to which the supervisor has adequate situation awareness.

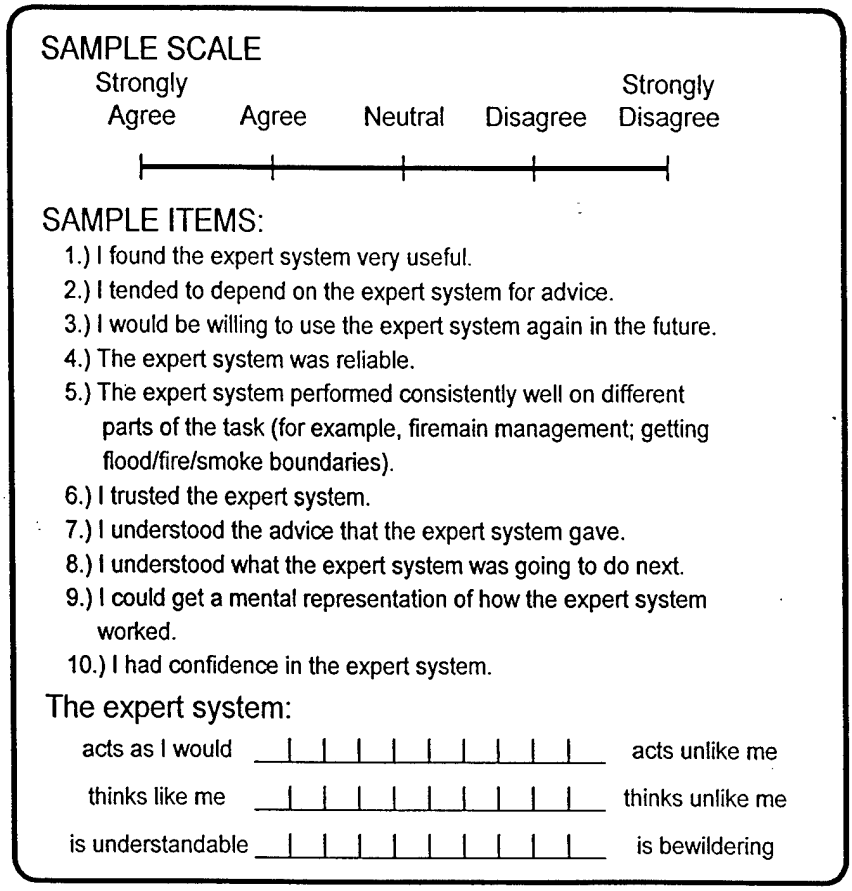


Figure 51. Possible items for inclusion in the STS instrument to measure supervisor trust

In summary, if the Automated Situation Awareness System cannot be shown to be accepted by damage control personnel on ship, it faces severe implementation problems. Yet it is important that supervisors do not become overly trusting of the system and fail to intervene as necessary. Thus, a goal is to obtain empirical data demonstrating a level of trust that is neither too high nor too low.

3.7.1.2 Measurement of Trust

The primary measure of trust will be a brief set of items forming what will be called the System Trust Scale (STS). Numerous items covering the attributes of trust listed in the preceding section will be developed from past research on trust [e.g., Van Swol and Sniezek, 1996]. The STS will be unique, however, in that it will attempt to capture the undesirable elements of over trust as well as mistrust. Examples of possible items for the STS are given in Figure 51. The STS will consist of items selected to meet standards of clarity and reliability. It will be administered to participants in all tests of the automated system at the ex-USS *Shadwell*.

Additional measures of trust are feasible. One would come from analysis of verbal protocols obtained during design tests for content related to issues of user trust in the system. Similarly, evidence of trust can be derived from log entries made by users in trial tests, as well as from interviews with users following experience with the components of the system. Indirect

evidence of trust in the system can be found in the behavioral data. For example, a low response threshold for over-ride decisions indicates a low level of trust in the system, while a high threshold suggests over trust.

3.7.2 System testing

Of course, testing of components of the system (e.g., supervisory interface design) will be a continuing part of system development. Here we address only the full-scale tests of the system with representative US Navy crews on the ex-USS *Shadwell*:

- FY98. (Full scale evaluation of baseline damage control operations and manning using the Damage Control System (DCS),
- FY00. Demonstration of a survivable semi-automated awareness system (casualty characterization and situational assessment) and [This goal was not fully achieved.]
- FY01. Demonstration of a fully automated system (casualty characterization, situational assessment, and casualty response). [This goal was not achieved.]

3.7.2.1 Design and Methodology

This data collection is intended to provide information about various attributes of the automated system in successive test components of the system as they are developed. This approach makes it possible to document changes in system performance due to successive advances in automation.

Upon implementation of the automated sensors, evaluation data collected on the ex-USS *Shadwell* will permit verification that the sensors are functioning, and make it possible to determine the value added by incremental increases in automation. For example, the improvements in response speed and effectiveness attributable to the automated sensors could be determined by comparing data from this test to those observed in baseline performance tests on the ex-USS *Shadwell* prior to initiation of the Automated Situation Awareness System. Similarly, testing of the fully automated system would provide data on the full extent of improvement in damage control operations due to the full automation developed with the system. The solution approach design will make it possible to investigate changes in supervisory control decisions and user and expert reactions to the system as it becomes increasingly automated.

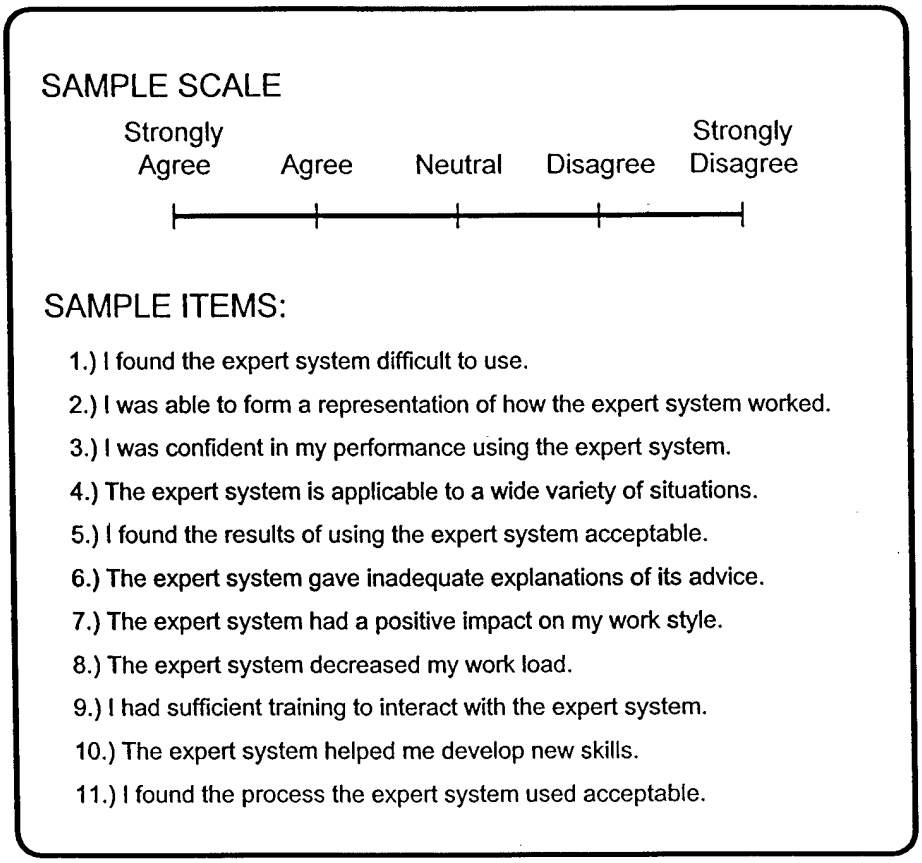


Figure 52. Example items for evaluation of the system

It should be noted that the evaluation of the final product of this effort will be most meaningful if an execution protocol is established prior to any testing, and followed for each test. The execution protocol refers to the precise nature and sequence of fire (instantaneous, incipient, growing, and flashover) and flooding, and is largely under the control of Navy personnel running the demonstrations and ex-USS *Shadwell* operations.

3.7.2.2 Evaluation Criteria

A first step is to identify the criteria on which the success of the system will be judged. Criteria expected to be important are the speed and effectiveness of performance by the automated system and supervisor, and subjective evaluations of this performance by users and experts. In accordance, several classes of measures are expected to be important:

1. Quantitative Physical Data—measures of physical variables describing the extent of secondary damage to the ship (e.g., variables related to secondary damage as presented in Sections 3.3.2 - 3.3.5. These would include readings at specified time intervals as well as the conclusion of damage control intervention.
2. Response Time—measures of elapsed time between critical events, such as initiation of a fire threat (instantaneous, incipient, growing, and flashover) or flooding, and detection, assessment, reaction, and control.

3. Supervisory Control Decisions—measures of the speed and quality of decisions regarding intervention with the automated system. This will include diagnosis of supervisor situation awareness and comprehension of system reasoning underlying actions.

4. Supervisor Assessment—measures of trust using instruments described above in Section 3.7.1, e.g., the Supervisor Trust Scale, and a measure to be based on modification of criteria given by Adelman, 1992, and Riedel and Pitz [1986]. (See sample items given in Figure 52)

5. Expert Assessment—systematic descriptions of perceived effectiveness of the system by damage control experts within the US Navy. Criteria will be adapted from Kumar [1990] and Mahmood and Sniezek, [1989].

It is assumed that the ex-USS *Shadwell* staff will be responsible for classes (1) and (2). Efforts will be made to develop measures and procedures for classes (3), (4), and (5) that avoid threats to validity [Campbell and Stanley, 1966; Cook and Campbell, 1979]. As feasible, evidence of reliability and validity will be obtained in accordance with guidelines given by Adelman [1992].

3.8 Major Task Interdependencies

A strength of our proposed solution approach is that the overall situation awareness and supervisory control task have been decomposed into 19 independent modules, all of which can be designed, developed, tested, and refined in parallel. Nevertheless, there are task demonstration interdependencies at the system level. These are shown in Figure 53.

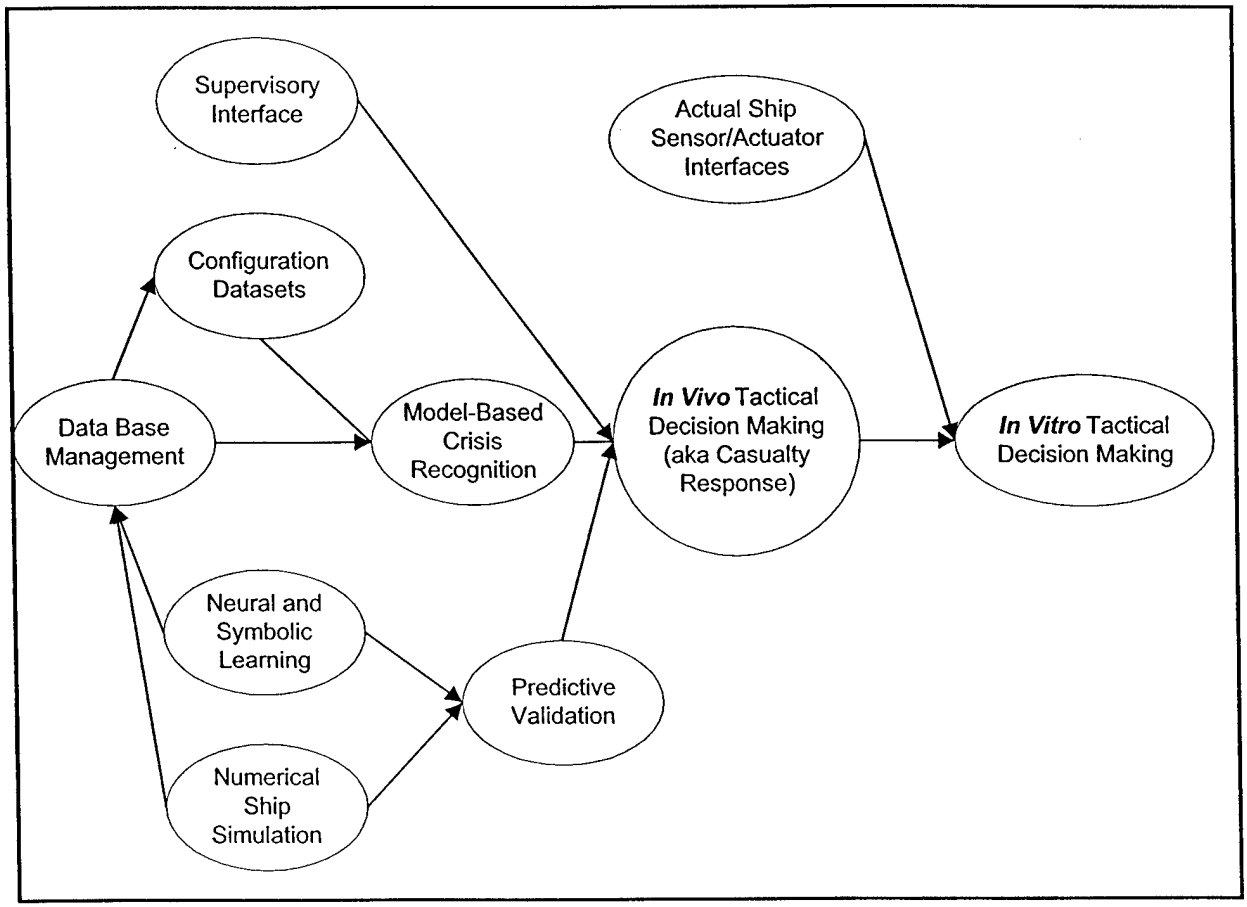


Figure 53. Major task interdependencies

The following provides some clarification of what is encompassed by some of the tasks shown in the task interdependency Figure above.

Database Management - The challenge of this activity is finding a database structure that can allow the exact same ship data to be used for three different purposes: Supervisory Interface/Visualization, Numerical Ship Simulation, and Intelligent Reasoning. Activities involved in this endeavor include:

- Incorporation of existing static and dynamic databases
- Manual knowledge acquisition of model-based assessment and casualty response
- Refinement of knowledge base using automated learning methods
- Temporal and deductive reasoning over the ground level relational database

- Configuration datasets for sensors

Crisis Recognition (a.k.a. Model Based Assessment) and Casualty Response (a.k.a. Tactical Decision Making).

- Knowledge base systems approach that integrates opportunistic blackboard architecture with neural network and belief network.

Model Based Prediction

- This encompasses Ship Simulation and Machine Learning
- Requires ship simulation modules, machine learning modules, intelligent object module, and scenario specification module.

Supervisory/Operator and other Interfaces

- Supervisory/Operator interface allows reasoning of system to be observed and selective overriding of automated system. A real or mock up version of this Supervisor Interface will be tested with experts each year.
- System interfaces are relevant to all major tasks

4. Progress to Date Relevance of Illinois DCA Scenario Generator Project

As a consequence of our creating the Illinois Damage Control Scenario Generator we have designed and/or have initial demonstrable working prototypes of 12 of the 19 modules that overlap between these efforts and the Automated Situation Awareness System.

The modules that have demonstrable initial working versions are:

- Ship Specification Module
- Scenario Specification Module
- Graphic Visualization Module
- SQL Static and Dynamic Database Module
- Deductive and Temporal Reasoning Module
- Simulation Engine
- Primary Damage Module
- Secondary Damage: Fire and Smoke Module
- Web Interface Module
- Secondary Damage: Flooding and Rupture Module
- Executive Control Module
- Intelligent Ship Objects

The modules that we would need to construct from scratch are the following:

- Supervisory Interface Module
- Ship Sensors/Actuator Interface Module
- Secondary Damage: Stability Module
- Predictive Validation System
- Opportunistic Blackboard Module
- Belief Network Module
- Neural Network Module

This same information is given better organization and detail in Figures 54-57.

Subsystem #1: HCI/Ship Interfaces	Illinois Scenario Generator	Automated Situation Awareness System
1. Ship Specification	Specify ship and its contents.	**SAME.** Extensions.
2. Scenario Specification	Specify a crisis scenario.	**SAME.** Extensions
3. Supervisory Control	Console interface for DCA.	**NEW.**
4. Graphic Visualization	3D Visualization of Crisis	**SAME.** Extensions.
5. Web Interface	Run system over Web.	**SAME.**
6. Ship Interface	---	**NEW.**

Figure 54. Degree of overlap between Illinois Scenario Generator and Automated Situation Awareness System for Subsystem #1

Subsystem #2: Physical Ship Simulation	Illinois Scenario Generator	Automated Situation Awareness System
1. Primary Damage	Create crisis.	**SAME.**
2. Fire and Smoke	Simulate fire and smoke spread	**SAME.**
3. Flooding and Rupture	Simulate flooding and rupture.	**SAME.**
4. Stability	---	**NEW.**
5. Simulation Engine	Executive control of simulation.	**SAME.**

Figure 55. Degree of overlap between Illinois Scenario Generator and Automated Situation Awareness System for Subsystem #2

Subsystem #3: Total Ship Representation	Illinois Scenario Generator	Automated Situation Awareness System
1. SQL Database (Static and Dynamic)	All Static and Dynamic Data.	**SAME.**
2. Deductive and Temporal Reasoning	Reasoning over database.	**SAME.**
3. Executive Control of all modules	Orchestrates all modules.	**SAME.** Extensions.

Figure 56. Degree of overlap between Illinois Scenario Generator and Automated Situation Awareness System for Subsystem #3

Subsystem #4: Intelligent Reasoning	Illinois Scenario Generator	Automated Situation Awareness System
1. Opportunistic Blackboard	---	**NEW.**
2. Neural Networks	---	**NEW.**
3. Belief Networks	---	**NEW.**
4. Predictive Crisis Validation	---	**NEW.**
6 Intelligent Ship Objects	Intelligent Ship Objects.	**SAME.** Extensions.

Figure 57. Degree of overlap between Illinois Scenario Generator and Automated Situation Awareness System for Subsystem #4

5. Related Research

5.1 Intelligent Reasoning Systems

The common interests of the decision sciences and artificial intelligence (AI) have been noticed in recent years [Horvitz et al., 88] [Henrion et al., 91]. Essentially, this interest can be expressed as reasoning under action given incomplete information and scarce resources. The decision sciences consist of Bayesian decision theory, the psychology of judgment, and their application in operations research and decision analysis. Decision theory is made up of probability theory and utility theory. Decision analysis is concerned with elicitation, representation, reasoning, and search procedures (one could therefore argue that decision-analytical AI is a more descriptive phrase than decision-theoretical AI, but we will use the latter term since it has been preferred in the literature).

What does decision theory have to offer AI? Decision-theoretic techniques offer a principled way to (i) reason under uncertainty and incompleteness; (ii) take into account complex preferences; and (iii) reason about decisions. Decisions underlie actions that an agent may take, and as AI moves toward complex, real-world problems, uncertainty becomes a prominent feature. Examples of using decision-theory in AI are the following. First, the notion of attitude toward risk has been discussed extensively within decision theory, and this may play a key role in real-world AI systems. Second, decision theory allows for trade-offs between different goals, in that goals are modeled as utility functions rather in a binary fashion as in traditional AI planning systems.

Decision theory, and in particular its subset probability theory, has recently been applied in several areas of AI. These include expert systems, planning, machine learning, control of inference, perception, model construction, temporal reasoning, nonmonotonic reasoning, intelligent tutoring, and natural language processing. In the following, we focus on the area of decision-theoretic expert systems, since this is historically the area where this line of research has been most prominent. In addition, this area is particularly relevant to the Automated Situation Awareness system.

Decision-theoretic (or normative) expert systems vary among the earliest expert systems. Early diagnostic expert systems, developed during the 1960s, were based on probability theory. The following simplifying assumptions were adopted in these early expert systems: First, the hypotheses (diseases) are mutually exclusive and collectively exhaustive. Second, any piece of evidence (symptom) is conditionally independent of any other piece of evidence. Let H be a hypothesis, and Ei and Ej two pieces of evidence. Then the conditional independence assumption amounts to letting P(Ei | H) = P(Ei | H, Ej). These two assumptions lead to simpler belief propagation and probability specification. Under the two assumptions, only *mn* conditional probabilities and *n-1* prior probabilities are needed, where *m* is the number of pieces of evidence and *n* is the number of hypotheses. Both evidence and hypotheses are assumed to be binary. These early expert systems performed at the level of experts or better, but their acceptability did not match their performance.

During the 1970s, the rule-based paradigm became an alternative to the decision-theoretic paradigm. Two well-known rule-based expert systems, developed during the 1970s, that incorporate uncertainty are MYCIN and PROSPECTOR. The uncertainty calculations in

these and similar expert systems were regarded as approximations to the probabilistic ideal, which was regarded as unattainable. However, the rule-based approach to reasoning under uncertainty also had its problems. The areas where rule-based inference proved to be most problematic were in the treatment of prior beliefs and in the assumption of modularity [Horvitz et al., 88]. Furthermore, there have been empirical studies showing that the uncertainty calculations used in an expert system do matter when evidence is weak or conflicting. Thus, there are both theoretical and practical reasons to consider alternatives.

The inherent weaknesses of the rule-based approach led to a renewed interest in decision theory in the 1980s and 1990s. Current research on decision theory for expert systems can be split into knowledge representation, knowledge engineering (or modeling), belief propagation, and explanation [Horvitz et al., 88]. Regarding knowledge representation, the notion of a decision basis is a complete representation of a decision problem. A decision basis consists of components representing alternatives, states, preferences, and relationships in a decision situation. Numerous representations of a decision basis have been developed. We will focus on influence diagrams, which is a graphical representation of the decision basis. An influence diagram is a directed acyclic graph (DAG) where the nodes represent properties or quantities of interest, arcs represent influence between nodes. An influence diagram consists of decision nodes, chance nodes, and value nodes. An influence diagram with only chance nodes is a belief network.

6. Summary

This report has presented a state-of-the art concept of automated situation awareness for ship damage control. The solution encompasses model-based crisis recognition, model-based predictive validation, automated casualty response, and a supervisor interface console.

7. Acronyms Used

ALARM	A Logical Alarm Reduction Mechanism
ALVINN	Autonomous Land Vehicle in a Neural Net
ANN	Artificial Neural Net
CAD	Computer Aided Design
CFAST	Consolidated Fire and Smoke Transport Model
CLT	Computational Learning Theory
D-CAMS	Damage Control Management Assets
DAG	Directed Acyclic Graph
DC	Damage Control
DC-ARM	Damage Control: Automation for Reduced Manning
DCA	Damage Control Assistant
DCAP	Damage Control Assistant Performance
DCS	Damage Control System
DDN	Dynamic Decision Network
ECL	Event Communication Language
EM	Electro-Mechanical Systems
EVDI	Expected Value of Displayed Information
EVRI	Expected Value of Revealed Information
FM	Fire Main
GUI	Graphical User Interface
H|C|F|E|D	Horizontal, Ceiling, Floor, Engulfment, Destruction
IDCTT	Integrated Damage Control Team Trainer
ISMS	Integrated Survivability Management System
KBE	Knowledge Base Editor

MBPS	Megabits Per Second
ODBC	Open Database Connectivity
PNP	Patch and Plug
RA	Research Assistant
RLL	Repair Locker Leader
SA	Situation Awareness
SGM	Scenario Generation Model
SQL	Structured Query Language. (Also Microsoft's relational database system)
SRSGM	Ship Representation Scenario Generation Module
STS	System Trust Scale
TMM	Time Map Maintenance
TRM	Temporal Resource Manager
TSR	Total Ship Representation
TSS	Total Ship Survivability
VLS	Vertical Launch System

8. References

Adams, M. J., Tenney, Y. J., & Pew, R. W. (1995). Situation awareness and the cognitive management of complex systems. *Human Factors, 37*(1), 85-104.

Adelman, L. (1992). *Evaluating Decision Support and Expert Systems.* New York: John Wiley & Sons, Inc.

Allen, J.F. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM, 26*, 832-843.

Anderson, J. R. (1989). The expert module. In Polson, M. C., & Richardson, J. J. (Eds.), *Foundations of Intelligent Tutoring Systems* (pp. 21-53). Hillsdale, MA: Lawrence Erlbaum Associates.

Anderson, J. R. (1990). Development of expertise. In Anderson, J. R., *Cognitive Psychology and Its Implications*. New York: W. H. Freeman and Company.

Arvey, R. D., Cole, D. A., Hazucha, J. F., & Hartanto, F. M. (1985). Statistical power of training evaluation designs. *Personnel Psychology, 38,* 493-507.

Backman, L., & Molander, B. (1991). On the generalizability of age-related decline in coping with high arousal conditions in a precision sport: Replication and extension. *Journal of Gerontology, 46*(2) 79-81.

Bacon, S. J. (1974). Arousal and the range of cue utilization. *Journal of Experimental Psychology, 102*(1), 81-87.

Barber, B. (1983). *The Logic and Limits of Trust.* New Brunswick: Rutgers University Press.

Basye, K., Dean, T., Kirman, J., & Lejter, M. (1992). A decision-theoretic approach to planning, perception, and control. *IEEE Expert, 7*(4), 58-65.

Baumann, M. R., Donovan, M. A., & Sniezek, J. A. (1997). Anxiety, positive thinking, on/off task thinking, and performance. Poster submitted to SIOP.

Baumann, M. R., Sniezek, J. A., Donovan, M. A., & Wilkins, D. C. (1996). *Evaluation of the Integrated Damage Control Training Technology* (Technical Report UIUC-BI-KBS-96008). Urbana: University of Illinois at Urbana-Champaign, Knowledge-Based Systems Group.

Blum, M. L., & Naylor, J. C. (1968). *Industrial Psychology: Its Theoretical and Social Foundations*. New York: Harper & Row.

Bowen, J. V. (1984). Computer assisted risk evaluation: A practical application to risk management decisions using engineering judgment. *Fire Safety Journal, 9*(2), 205-210.

Bratley, P., Fox, B. L., & Schrage, L. E. (1983). *A Guide to Simulation*. New York: Springer-Verlag.

Breglia, D. R., & Fowlkes, D. H. (1991). Virtual environment training technology. In *Proceedings of the 1991 International Simulation Conference* (pp. 204-209). San Diego, CA: Society of Computer Simulation.

Brown, D., & Chandrasekaran, B. (1986). Knowledge and control for a mechanical design expert system. *IEEE Computer, 19(7),* 92-100.

Buchanan, B. G., & Smith, R. G. (1989). Fundamentals of expert systems. In Barr, A., Cohen, P. R., & Feigenbaum, E. A. (Eds.), *The Handbook of Artificial Intelligence*: *Vol. IV* (pp. 149-192). New York: Addison-Wesley.

Buchanan, B. G., & Wilkins, D. C. (1993). *Readings in Knowledge Acquisition and Learning*: *Automating the Construction and Refinement of Expert Systems*. San Mateo, CA: Morgan Kaufmann Publishing.

Burton, D. (1988). Do anxious swimmers swim slower? Reexamining the elusive anxiety performance relationship. *Journal of Sport Psychology, 10,* 45-61.

Burton, R., & Brown, J. S. (1979). An investigation of computer coaching for informal learning activities. *International Journal of Man-Machine Studies, 11,* 5-24.

Campbell, D. T., & Stanley, J. C. (1966). *Experimental and Quasi-Experimental Design for Research*. Chicago: Rand McNally.

Cannon-Bowers, J. A., Salas, E., Tannenbaum, S. I., & Mathieu, J. E. (1995). Toward theoretically based principles of training effectiveness: A model and initial empirical investigation. *Military Psychology, 7*(3), 141-164.

Chandrasekaran, B. (1986). Generic tasks in knowledge-based reasoning: High-level building blocks for expert system design. *IEEE Expert, 1*(3), 23-29.

Charniak, E. (1991). Bayesian networks without tears. *AI Magazine, 12*(4), 50-63.

Charniak, E., & Goldman, R. P. (1993). A Bayesian model of plan recognition. *Artificial Intelligence, 64*(1), 53-79.

Chi, M. T. H., Glaser, R., & Farr, M. J. (1988). *The Nature of Expertise*. Hillsdale, MA: Lawrence Erlbaum Press.

Clancey, W. J. (1984). NEOMYCIN: Reconfiguring a rule-based system with application to teaching. In Clancey, W. J., & Shortliffe, E. H. (Eds.), *Readings in Medical Artificial Intelligence* (pp. 361-381). Reading, MA: Addison-Wesley.

Clancey, W. J. (1985). Heuristic classification. *Artificial Intelligence, 27,* 289-350.

Clancey, W. J. (1987). *Knowledge-based tutoring: The Guidon Program.* Cambridge, MA: MIT Press.

Clancey, W. J. (1988). Acquiring, representing, and evaluating a competence model of diagnosis. In Chi, M. T. H., Glaser, R., & Farr, M. J. (Eds.), *The Nature of Expertise* (pp. 343-418). Hillsdale, MA: Lawrence Erlbaum Press.

Clancey, W. J. (1992). Model construction operators. *Artificial Intelligence, 53*(1), 1-115.

Cohen, M. S., & Freeman, J. T. (1996). Metarecognition in time-stressed decision making: Recognizing, critiquing, and correcting. *Human Factors, 38,* 206-219.

Conlon, T. M. (1992). *DDG-51 operational evaluation: Measures of workload from combat information center communication patterns.* Unpublished master's thesis. Naval Postgraduate School, Monterey, California.

Cook, T. D., & Campbell, D. T. (1979). *Quasi-Experimentation: Design & Analysis Issues for Field Settings.* Boston: Houghton Mifflin Company.

Cover, T. M., & Thomas, J. A. (1991). *Elements of Information Theory.* New York: John Wiley and Sons.

Cox, G. (1995). *Combustion Fundamentals of Fire.* London: Academic Press.

Dean, T., & Boddy, M. (1988). An analysis of time-dependent planning. In *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88)* (pp. 21-26). San Mateo, CA: Morgan Kaufmann.

Dean, T., & Kanazawa, K. (1989). A model for reasoning about persistence and causation. *Computational Intelligence, 5*(3), 142-150.

Dean, T. L., & McDermott, D.V. (1987). Temporal database management. *Artificial Intelligence, 32,* 1-55.

Deitenberger, M. A. (1989). *A Validated Furniture Fire Model with FAST (HEMFAST)* (NIST-GCR-89-564). Gaithersburg: National Institute of Standards and Technology.

Derosier, P. (1994). *Integrated Damage Control Training Technology: System Manager Manual.* Bethesda: Center for Interactive Media in Medicine.

Donoho, S., & Wilkins, D. C. (1994a). Odysseus2: Addressing the challenges of apprenticeship. In *Knowledge Acquisition for Knowledge-Based Systems Workshop* (pp. 14.1-14.8). Banff, Canada.

Donoho, S., & Wilkins, D. C. (1994b). Exploiting the ordering of observed problem-solving steps for knowledge base refinement. In *Proceedings of the Twelfth National Conference on Artificial Intelligence* (pp. 569-575). Menlo Park, CA: AAAI Press.

Driskell, J. E., & Salas, E. (1991). Group decision making under stress. *Journal of Applied Psychology, 76*(3), 473-478.

Driskell, J. E., & Salas, E. (1996). *Stress and Human Performance.* Mahwah, NJ: Lawrence Erlbaum Associates.

Duchastel, P. (1991). Towards methodologies for building knowledge-based instructional systems. *Instructional Science, 20,* 349-358.

Durkin, A. F., Williams, F. W., Schaffer, J. L., Toomey, T. A., Hunt, S. P., & Darwin, R. L. (1993). *Post-flashover fires in shipboard compartments aboard ex-USS Shadwell: Phase IV - Impact of Navy Fire Insulation.* NRL Memorandum Report NRL/MR/6183-93-7335, 9 June 1993

Durkin, J. (1994a). *Expert Systems: Catalogue of Applications.* Akron: Intelligent Computer Systems Press.

Durkin, J. (1994b). *Expert Systems: Design and Development.* New York: Macmillan Publishing.

Durlach, N. L., Pew, R. W., Aviles, W. A., DiZio, P. A., & Zeltzer, D. L. (1992). *Virtual Environment Technology for Training (VETT)* (BBN Report No. 7661). Cambridge, MA: Bolt, Beranek, & Newman, Inc.

Earle, T. C., & Cvetkovich, G. T. (1995). *Social Trust.* London: Praeger.

Easterbrook, J. A. (1959). The effect of emotion on cue utilization and the organization of behavior. *Psychological Review, 66,* 183-201.

Edland, A., & Svenson, O. (1993). Judgment and decision making under time pressure: Studies and findings. In Svenson, O. & Maule, A. J. (Eds.), *Time Pressure and Stress in Human Judgment and Decision Making* (pp. 27-40). New York: Plenum Press.

Elstein, A. S., Shulman, L. S., & Sprafka, S. A. (1978). *Medical Problem Solving: An Analysis of Clinical Reasoning.* Cambridge, MA: Harvard University Press.

Endsley, M. R. (1995). Measurement of situation awareness in dynamic systems. *Human Factors, 37*(1), 65-84.

Ericcson, K. A., & Simon, H. (1984). *Protocol Analysis: Verbal Reports as Data.* Cambridge, MA: MIT Press.

Ericcson, K. A., & Smith, J. (1991a). *Towards a General Theory of Expertise: Prospects and Limits.* Cambridge, UK: Cambridge University Press.

Ericcson, K. A., & Smith, J. (1991b). Prospects and Limits of the Empirical Study of Expertise. Introduction to Ericcson, K. A., & Smith J., *Towards a General Theory of Expertise* (pp. 1-39). Cambridge, UK: Cambridge University Press.

Ericcson, K. A., Krampe, R. T., & Tesch-Romer, C. (1993). The role of deliberate practice in the acquisition of expert performance. *Psychological Review, 100*(3), 363-407.

Erman, L. D., London, P. E., & Fickas, S. F. (1981). The design and an example of use of HEARSAY-III. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence* (pp. 409-415). Vancouver, British Columbia.

F.C.S.L., C.I.S.E., P.O.L.I.M.I., I.C., E.P.F.L., U.C.L., (ESPIRIT P2409 Consortium), EQUATOR: Ontology of Processes, Events, and Time in Industrial Information Processing Systems. *Document, Product: D211-1.* Feb 1990.

Feigenbaum, E. A. (1993). Tiger in a cage: The applications of knowledge based systems. In *Proceedings of the Eleventh National Conference on Artificial Intelligence* (p. 852).

Feltovich, P. J., Johnson, P. E., Moller, J. H., & Swanson, D. B. (1984). LCS: The role and development of medical knowledge in diagnostic expertise. In Clancey,

W. J., & Shortliffe, E. H. (Eds.), *Readings in Medical Artificial Intelligence* (pp. 275-319). Reading, MA: Addison-Wesley.

Fischer, G., Lemke, A. C., & Mastaglio, T. (1991). Critics: An emerging approach to knowledge-based human-computer interaction. *International Journal of Man-Machine Studies, 35*(5), 695-721.

Fletcher, J. D. (1992). *Courseware Portability* (IDA Paper P-2648). Institute for Defense Analyses.

Fu, L. (1994a). Knowledge-Based Neural Networks. In Fu, L., *Neural Networks in Computer Intelligence.* New York: McGraw-Hill.

Fu, L. (1994b). Rule Generation From Neural Networks. In Fu, L., *Neural Networks in Computer Intelligence.* New York: McGraw-Hill.

Furst, D. M., & Tenenbaum, G. (1985). Influence of attentional focus on reaction time. *Psychological Reports, 56,* 299-302.

Gaba, D. M., & DaAnda, A. (1988). A comprehensive anesthesia simulation environment: Re-creating the operating room for research and teaching. *Anesthesiology, 69,* 387-394.

Ghoniem, A. F., Marek, C. J., & Oppenheim, A. K. (1983). *Modeling interface motion of combustion: A computer code for two-dimensional, unsteady turbulent combustion* (NASA Technical Paper 2132). NASA, Scientific and Technical Information Branch.

Giarratano, J., & Riley, G. (1994). *Expert Systems: Principles and Programming* (2nd ed.). Boston: PWS Publishing.

Gillie, T., & Broadbent, D. (1989). What makes interruptions disruptive? A study of length, similarity, and complexity. *Psychological Research, 50,* 243-250.

Goldman, R. P., & Charniak, E. (1993). A language for construction of belief networks. *IEEE Transactions on Pattern Analysis & Machine Intelligence, 15*(3), 196-207.

Goodman, B., & Litman, D. (1990). Plan recognition for intelligent interfaces. In *Proceedings of the 1990 IEEE International Conference of Computer Design: VLSI in Computers and Processors* (pp. 297-303).

Gruber, T. G. (1989). Automated knowledge acquisition for strategic knowledge. *Machine Learning, 4,* 293-336.

Haessler, W. M. (1989). *Fire: Fundamentals and Control.* New York: Marcel Dekker, Inc.

Hamblin, R. L. (1958). Leadership and crisis. *Sociometry, 21,* 322-335.

Hardy, L., & Parfitt, G. (1991). A catastrophe model of anxiety and performance. *British Journal of Psychology, 82*(2), 163-178.

Hassoun, M. H. (1996). *Fundamentals of Artificial Neural Networks.* Cambridge, MA: MIT Press.

Hayes-Roth, B. (1995a). An Architecture for Adaptive Intelligent Systems. *Artificial Intelligence: Agents and Interactivity* [special issue], *72,* 329-365.

Hayes-Roth, B. (1995b). *Guardian Project Home Page.* URL: http://www-ksl.stanford.edu /projects/guardian/index.html.

Hayes-Roth, B., Pfleger, K., Lalanda, P., Morignot, P., & Balabanovic, M. (in press). A domain-specific software architecture for adaptive intelligent systems. *IEEE Transactions on Software Engineering.*

Hayes-Roth, B., Washington, R., Ash, D., Hewett, R., Collinot, A., Vina, A., & Seiver, A. (1992). Guardian: A prototype intelligent agent for intensive-care monitoring. *Journal of Artificial Intelligence in Medicine, 4,* 165-185.

Henrion, M., Breese, J. S., & Horvitz, E. J. (1991). Decision analysis and expert systems. *AI Magazine, 12*(4), 64-90.

Hogan, J., Petersons, A. V., Salas, E., & Reynolds, R. E. (1991). *Team Performance, Training Needs and Teamwork: Some Field Observations.* Orlando, FL: Naval Training Systems Center.

Hogge, John C. (1987a). *TIME and TPLAN User's Manual* (Tech Report No. UIUCDCS-R-87-1366). Urbana: University of Illinois at Urbana-Champaign, Department of Computer Science.

Hogge, John C. (1987b). *TPLAN : A Temporal Interval-Based Planner with Novel Extensions* (Tech Report No. UIUCDCS-R-87-1367). Urbana: University of Illinois at Urbana-Champaign, Department of Computer Science, Department of Computer Science.

Hollnagel, E. (1987). Commentary: Issues in knowledge-based decision support. *International Journal of Man-Machine Studies, 27,* 743-751.

Holzman, R. S., Cooper, J. B., Gaba, D. M., Philip, J. H., Small, S., & Feinstein, D. (in press). Anesthesia crisis resource management: Real-life simulation training in operating room crises. *Journal of Clinical Anesthesia.*

Horvitz, E. J., Breese, J. S., & Henrion, M. (1988). Decision theory in expert systems and artificial intelligence. *International Journal of Approximate Reasoning, 2,* 247-302.

Howard, S. K., Gaba, D. M., Yang, G. S., & Sarnquist, F. H. (1992). Anesthesia crisis resource management training: Teaching anesthesiologists to handle critical incidents. *Aviation, Space, and Environmental Medicine, 63,* 763-770.

Isenberg, D. J. (1981). Some effects of time-pressure on vertical structure and decision-making accuracy in small groups. *Organizational Behavior and Human Performance, 27*(1), 119-134.

Jelinek, F. (1968). *Probabilistic Information Theory.* New York: McGraw-Hill.

Jensen, F. V. (1996). *An Introduction to Bayesian Networks.* New York: Springer-Verlag.

Johansson, G. (1989). Stress, autonomy, and the maintenance of skill in supervisory control of automated systems. *Applied Psychology: An International Review, 38,* 45-56.

Johnson, M. (1994). *Validation of an active multimedia courseware package for the integrated damage control training technology trainer.* Unpublished master's thesis. Naval Postgraduate School, Monterey, California.

Johnson, P. E., Duran, A. S., Hassebrock, F., Moller, J., Prietula, M., Feltovich, P. J., & Swanson, D. B. (1981). Expertise and error in diagnostic reasoning. *Cognitive Science, 5,* 235-283.

Jones, W. W. (1985). A multi-compartment model for the spread of fire, smoke, and toxic gases. *Fire Safety Journal, 9*(3), 55-79.

Kelly, V. (1984). The CRITTER system: Automated critiquing of digital circuit designs. In *Proceedings of the 21st Design Automation Conference* (pp. 419-425).

Klein, G. (1996). The effect of acute stressors on decision making. In Driskell, J. E., & Salas, E. (Eds.), *Stress and Human Performance*. Mahwah, NJ: Lawrence Erlbaum Associates.

Koomen, J. A. G. M. (1989a). *The TIMELOGIC temporal reasoning system.* (CS Department Technical Report No. 231). Rochester, NY: University of Rochester, Department of Computer Science.

Koomen, J. A. G. M. (1989b). *Reasoning about recurrence.* (CS Department Technical Report No. 307). Rochester, NY: University of Rochester, Department of Computer Science.

Kotteman, J. E., Davis, F. D., & Remus, W. E. (1994). Computer-aided decision making: Performance, beliefs, and the illusion of control. *Organizational Behavior and Human Decision Processes, 57,* 26-37.

Kowalski, T., & Thomas, D. (1985). The VLSI design automation assistant: What's in a knowledge base? In *Proceedings of the 22nd Design Automation Conference* (pp. 252-258).

Kumar, K. (1990). Post implementation evaluation of computer-based information systems: Current practices. *Communications of the ACM, 33*(2), 203-212.

Langlotz, C. P., & Shortliffe, E. H. (1983). Adapting a consulting system to critique user plans. *International Journal of Man-Machine Studies, 19,* 479-496.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten ZIP code recognition. *Neural Computation, 1*(4), 541-551.

Lee, J. D., & Neville, M. (1994). Trust, self-confidence, and operators' adaptation to automation. *International Journal of Human-Computer Studies, 40,* 153-184.

Leon, M. R., & Revelle, W. (1985). The effects of anxiety on analogical reasoning: A test of three theoretical models. *Journal of Personality and Social Psychology, 49,* 1302-1315.

Lesgold, A. M. (1984). Acquiring expertise. In Anderson, J. R., & Kosslyn, S. M. (Eds.), *Tutorials in Learning and Memory: Essays in Honor of Gordon Bower* (pp. 31-60). San Francisco: Freeman.

Lesgold, A. M. (1993). Ideas about feedback and their implications for intelligent coached apprenticeship. *MML* 1/94.

Lesgold, A. M., Kotz, S., Greenberg, L., Hughes, E., & Eggan, G. (1991). Intelligent coaches apprenticeship systems: Experiences from the Sherlock project. In *Proceedings of the 1991 International Conference on Systems, Man and Cybernetics.* Charlottesville, NC.

Lestina, T., Runnerstrom, E., Davis, K., Durkin, A. and Williams, F.W., "Evaluation of Firemain Architectures and Supporting Reflexive Technology," NRL Memorandum Report NRL/MR 6180—99—8346, March 12, 1999

Ling, W. C. T., & Williamson, R. B. (1985). Modeling of fire spread through probabilistic networks. *Fire Safety Journal, 9*(1), 287-300.

Little, M. C. (1995). *C++Sim Simulation Class Library Home Page.* URL: http: //ulgham.ncl.ac.uk/C++SIM/homepage.html.

Locke, E. A., & Latham, G. P. (1984). *A Theory of Goal Setting and Task Performance.* Englewood Cliffs, NJ: Prentice-Hall.

Lotvin, M., et al., (1984). AMOEBA: a symbolic VLSI layout system. In *Proceedings of the 21st Design Automation Conference* (pp. 294-300).

Madni, A. M. (1988). The role of human factors in expert systems design and acceptance. *Human Factors, 30*(4), 395-414.

Mahmood, M., & Sniezek, J. A. (1989). Defining decision support systems: An empirical assessment of end-user satisfaction. *INFOR: Canadian Journal of Operations Research and Information Processing, 27*(3), 253-271.

Maiocchi, R., & Pernici, B. (1988). *Temporal data management in real-time systems: A comparative view* (Internal Report No. 88-046). Milan: Department di Elettronica, Politecnico di Milano.

Maule, J.A., & Svenson, O. (1993). Concluding remarks. In Svenson, O., & Maule, A. J. (Eds.), *Time Pressure and Stress in Human Judgment and Decision Making* (pp. 323- 329). New York: Plenum Press.

Mengshoel, O. J., & Wilkins, D. C. (1996). Recognition and critiquing of erroneous agent actions. In Tambe, M., & Gmytrasiewicz, P. (Eds.), *AAAI-96 Workshop on Agent Modeling* (pp. 61-68). Portland, OR: AAAI Press.

Miller, J. R. (1988). The role of human-computer interaction in intelligent tutoring systems. In Polson, M. C., & Richardson, J. J. (Eds.), *Foundations of Intelligent Tutoring Systems,* (pp. 143-189). Hillsdale, NJ: Lawrence Erlbaum Associates.

Miller, P. L. (1986). *Expert Critiquing Systems: Practice-Based Medical Consultation by Computer.* New York: Springer-Verlag.

Molander, B., & Backman, L. (1994). Attention and performance in miniature golf across the life span. *Journal of Gerontology, 49*(2), 35-41.

Molloy, R., & Parasuraman, R. (1996). Monitoring an automated system for a single failure: Vigilance and task complexity effects. *Human Factors, 38,* 311-322.

Morgan, B. B., Gilckman, A. S., Woodward, E. A., Blaiwes, A. S., & Salas, E. (1986). *Measurement of Team Behaviors in a Navy Environment* (Technical Report No. 86-014). Navy Training Systems Center, Orlando, Florida.

Morris, L. W., Davis, M. A., & Hutchings, C. H. (1981). Cognitive and emotional components of anxiety: Literature review and a revised worry-emotionality scale. *Journal of Educational Psychology, 73*(4), 541-555.

Naval Research Laboratory (1992). *Review of fire spread parameters for total ship survivability/fleet training fire spread model.* (Memo 6180-107). Washington, DC

Naval Sea Systems Command (1992). *The integrated survivability management system: Shipboard damage control greets the 21st century.*

Neapolitan, R. E. (1990). *Probabilistic Reasoning in Expert Systems.* New York: John Wiley & Sons.

Nemerich, C., & Durking, J. (1995). *Operational objectives for real time stability status.* Washington, DC: Naval Sea Systems Command.

Newell, A. (1969). Heuristic programming: Ill-structured problems. In Aronofsky, J. (Ed.), *Progress in Operations Research* (pp. 360-414). New York: Wiley.

Nii, H. P. (1989). Blackboard systems. In Barr, A., Cohen, P. R., & Feigenbaum, E. A. (Eds.), *The Handbook of Artificial Intelligence: Vol. IV* (pp. 1-82). Reading, MA: Addison-Wesley.

Nii, H. P., & Feigenbaum, E. A. (1978). Rule-based understanding of signals. In Waterman, D. A. & Hayes-Roth, R. (Eds.), *Pattern-Directed Inference System* (pp. 483-501). New York: Academic Press.

Nii, H. P., Feigenbaum, E. A., Anton J. J., & Rockmore, A. J. (1982). Signal-to-symbol transformation: HASP/SIAP case study, *AI Magazine, 3*(2), 23-25.

Noon, R. (1995). *Engineering Analysis of Fires and Explosions*. Boca Raton, FL: CRC Press.

Olson, M. & Sniezek, J. A. (1996). *Overconfidence matters: The cost of action based on miscalibrated probability assessments for choices.* Manuscript submitted for publication.

Orasanu, J. M., & Backer, P. (1996). Stress and military performance: The effect of acute stressors on decision making. In Driskell, J. E., & Salas, E. (Eds.), *Stress and Human Performance*. Mahwah, NJ: Lawrence Erlbaum Associates.

Ousterhout, J., et al, (1984). Magic: A VLSI layout system. In *Proceedings of the 21st IEEE Design Automation Conference* (pp. 152-159).

Parasuraman, R., Molloy, R., & Singh, I. L. (1993). Performance consequences of automation-induced "complacency." *International Journal of Aviation Psychology, 3,* 1-23.

Park, Y. T., Donoho, S., & Wilkins, D. C., (1994). Recursive heuristic classification. *Journal of Expert Systems, 7*(4), 329-357.

Park, Y. T., Tan, K. W., & Wilkins, D. C., (1991) *MINERVA: A knowledge-based expert system shell with declarative representation and flexible control* (Report UIUC-KBS-90-017). Urbana: University of Illinois at Urbana-Champaign, Department of Computer Science.

Park, Y. T., Donoho, S., Tan, K. W., Mengshoel, O. J., & Wilkins, D. C. (1995). *MINERVA 3.0: A knowledge-based system shell with declarative representation and flexible control* (Technical Report UIUC-KBS-92-012). Urbana: University of Illinois at Urbana-Champaign, Department of Computer Science.

Patel, V. L., & Groen, G. J. (1991). The general and specific nature of expertise: A critical look. In Chi, M. T. H., Glaser, R., & Farr, M. J. (Eds.), *Towards a General Theory of Expertise* (pp. 93-125). Cambridge, UK: Cambridge University Press.

Pauker, S., Gorry, G., Kassirer, J., & Schwartz, W. (1976). Toward a simulation of clinical cognition: Taking the present illness by computer. *American Journal of Medicine, 60,* 981-995.

Peacock, R. D., Forney, G. P., Reneke, P., Portier, R., & Jones, W. W. (1993) *CFAST: The consolidated model of fire growth and smoke transport* (NIST Technical Note 1299). U. S. Department of Commerce, National Institute of Standards and Technology.

Peatross, M.J., Luers, A.C., Pham, H., Scheffey, J.C. Wong, J.T., Farley, J.P., Rose-Pehrsson S.L., Nguyen, X.U., Tatem, P.A. and Williams, F.W., "Results of FY 2000 DC-ARM Demonstration, NRL Ltr Rpt 6180/0029, 7 February 2001

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* San Mateo, CA: Morgan-Kaufmann.

Pohl, J., et al, (1992). *A Computer-Based Design Environment: Implemented and Planned Extensions of the ICADS Model.* California Polytechnic State University.

Pomerleau, D. A. (1991). Efficient training of artificial neural networks for autonomous navigation. *Neural Computation, 3*(1), 88-97.

Psotka, J. (1994). *Immersive Tutoring Systems: Virtual Reality and Education Training.* Alexandria, VA: Army Research Institute.

Radhakrishnan, P., Arrow, H. A., & Sniezek, J. A. (1996). Hoping, Performing, Learning, and Predicting: Changes in the Accuracy of Self-Evaluations of Performance. *Human Performance,* 9(1), 23-49.

Reddy, D. R., Erman, L. D., & Neely, R. B. (1973). The HEARSAY speech understanding system: An example of the recognition process. In *Proceedings of the 3rd International Joint Conference on Artificial Intelligence* (pp. 185-193). Stanford, CA: Stanford University.

Riedel, S. L., & Pitz, G. F. (1986). Utilization-oriented evaluation of decision support systems. *IEEE Transactions on Systems, Man and Cybernetics, 6*(16), 980-996.

Rook, F., & Donnell, M. (1993). Human cognition and the expert system interface: Mental models and inference explanations. *IEEE Transactions on Systems, Man, and Cybernetics, 23*(6), 1649-1661.

Rouse, W. B., Cannon-Bowers, J. A., & Salas, E. (1992). The role of mental models in team performance in complex systems. *IEEE Transactions on Systems, Man and Cybernetics, 22*(6), 1296-1308.

Ruffell-Smith, H. P. (1979). *A simulator study of the interaction of pilot workload with errors, vigilance, and decisions* (NASA TM-78483). Washington, DC: NASA.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning by internal representation by error propagation. In Rumelhart, D. E. (Ed.), *Parallel Distributed Processing Explorations in the Microstructure of Cognition, Vol. 1: Foundations.* Cambridge, MA: MIT Press.

Salas, E., Bowers, C. A., & Cannon-Bowers, J. A. (1995). Military team research: 10 years of progress. *Military Psychology, 7*(2), 55-75.

Salas, E., Burgess, K. A., & Cannon-Bowers, J. A. (1995). Training effectiveness techniques. In Weiner, J. (Ed.), *Research Techniques in Human Engineering* (pp. 439-471). Englewood, NJ: Princeton & Hall.

Salas, E., Driskell, J. E., & Hughes, S. (1996). The study of stress and human performance: The effect of acute stressors on decision making. In Driskell, J. E., & Salas, E. (Eds.), *Stress and Human Performance.* Mahwah, NJ: Lawrence Erlbaum Associates.

Sarter, N. B., & Woods, D. D. (1995). How in the world did we ever get into that mode? Mode error and awareness in supervisory control. *Human Factors, 37,* 5-19.

Schlenker, B. R., Helm, B., & Tedeschi, J. T. (1973). The effects of personality and situational variables on behavioral trust. *Journal of Personality and Social Psychology, 25,* 419-427.

Selman, B., Brooks, R. A., Dean, T., Horvitz, E., Mitchell, T. M., & Nilsson, N. J (1996). Challenge Problems for Artificial Intelligence, *AAAI96P*, 1340-1345.

Shavlik, J. W., & Geoffrey, G. T. (1989). An approach to combining explanation-based and neural learning algorithms. *Connection Science, 1*(3).

Shavlik, J. W., & Towell, G. G. (1989). An approach to combining explanation-based and neural learning algorithms. *Connection Science, 1*, 231-253.

Shaw, M. L. G., & Woodward, J. B. (1990). Modeling expert knowledge. *Knowledge Acquisition, 2*(2), 179-206.

Sheridan, T. B. (1992). *Telerobotics, Automation, and Human Supervisory Control.* Cambridge, MA: MIT Press.

Sherrie, G. P. (1988). Apprenticeship Instruction for Real-World Tasks: The Coordination of Procedures, Mental Models, and Strategies. *Review of Research in Education, 15,* 97-169.

Shoham, Y., & Goyal, N. (1988). Temporal reasoning in artificial intelligence. In Shrobe, H. (Ed.), *Exploring Artificial Intelligence* (pp. 419-438). San Mateo, CA: Morgan Kaufmann.

Silverman, B. G. (1990). *Human Error.* New York: Cambridge University Press.

Silverman, B. G. (1992a). *Critiquing Human Error: A Knowledge-Based Human-Computer Collaboration Approach.* New York: Academic Press.

Silverman, B. G. (1992b). Building a better critic: Recent empirical results. *IEEE Expert, 7*(2), 18-24.

Silverman, B. G., & Mezher, T. (1992). Expert critics in engineering design: Lessons learned and research needs. *AI Magazine, 13*(1), 45-62.

Simon, H. A. (1969). *The Sciences of the Artificial.* Cambridge, MA: MIT Press.

Sniezek, J.A. (1986). The role of variable labels in cue probability learning tasks. *Organizational Behavior and Human Decision Processes*, 38,141-161.

Sniezek, J.A. (1988). Prediction with single-event vs. aggregate data. *Organizational Behavior and Human Decision Processes*, 41(2),196-210.

Sniezek, J.A., & Henry, R. (1989). Accuracy and confidence in group judgment. *Organizational Behavior and Human Decision Processes*, 43(1), 1-28.

Sniezek, J. A. (1989). An examination of group process in judgmental forecasting. *International Journal of Forecasting*, 5, 171-178.

Sniezek, J. A., May, D., & Sawyer, J. E. (1990). Social uncertainty and interdependence: A study of resource allocation decisions in groups. *Organizational Behavior and Human Decision Processes*, 46(2), 155-180.

Sniezek, J. A., Paese, P. W., & Switzer, F. S., III (1990). The effect of choosing on confidence in choice. *Organizational Behavior and Human Decision Processes*, 46(2), 264-282.

Sniezek, J. A. (1990). A comparison of group techniques for prediction from shared information. *Group and Organization Studies*, 15(1), 5-19.

Sniezek, J.A., & Henry, R. (1990). Revision, weighting, and commitment in consensus group judgment. *Organizational Behavior and Human Decision Processes*, 45(1), 66-84.

Sniezek, J. A. and Buckley, T. (I991). Confidence depends on level of aggregation. *Journal of Behavioral Decision Making*, Vol. 4, 263-272.

Sniezek, J. A. (1992). Groups under uncertainty: An examination of confidence in group decision making. In James H. Davis (Ed.) special issue of Group Decision Making for *Organizational Behavior and Human Decision Processes*, 52, 124-155.

Sniezek, J. A., and Buckley, T. (1993). Becoming more or less uncertain.... In John Castellan (Ed.), *Current Issues in Individual and Group Decision Making*, Hillsdale, NJ: Lawrence Erlbaum and Associates.

Sniezek, J. A., Kuhn, K. & Spurlock, D. (1994). DGERM: The Decision Guide for Environmental Risk Management. *United States Army Construction Engineering Research Laboratory Technical Report.*

Sniezek, J. A. and Buckley, T. (1995). Choice accuracy and confidence in Judge Advisor Decision Making Systems. *Organizational Behavior and Human Decision Processes*, 62(2), 159-174.

Song, F. (1992). *Combining Temporal and Hierarchical Constraints for Temporal Reasoning.* [Available by request from fsong@uoguelph.ca]

Spickelmier, R., & Newton, A. (1988). CRITIC: A knowledge-based program for critiquing circuit designs. In *Proceedings of the 1988 IEEE International Conference of Computer Design: VLSI in Computers and Processors* (pp. 324-327).

Spielberger, C. D. (Ed.) (1972). *Anxiety: Current Trends in Theory and Research: Vol. 1.* New York: Academic Press.

Sticht, T., Ellis, J., Montague, W., Quellmalz, E., & Slappy, J. (1992). Combining environmental design and computer programs to enhance learning in Navy technical training. *Military Psychology, 5*(1), 63-75.

Tatem, P. A. (1994). *Proceedings of the Damage Control/Fire Fighting into the 21st Century Workshop.*

Tatem, P. A., Ahmed, G. N., Dietenberger, M. A., & Jones, W. W., *Calculating flame spread on horizontal and vertical surfaces,* NRL Memorandum Report, NRL/MR/6180-94-7493, 8 August 1994

Taylor, H. M. S. (1984). *An Introduction to Stochastic Modeling.* San Diego, CA: Academic Press.

Trafimow, D., and Sniezek, J. A. (1994). Perceived expertise and its effect on confidence. *Organizational Behavior and Human Decision Processes.* 57 (2), 290-302.

Van Swol, L. & Sniezek, J. A. (1996, August). *Monetary influence of trust in a judge-advisor system.* Poster session presented at the annual meeting of the Academy of Management, Cincinnati, OH.

Weaver, J. L., Morgan, B. B., Adkins-Holmes, C., & Hall, J. K. (1991). *A review of potential moderating factors in the stress/performance relationship.* (NAVTRASYSCEN Technical Report 92-012). Orlando, FL: Naval Training Systems Center.

Weicheng, F. (1991). *Computer Modeling of Combustion Processes.* Beijing, People's Republic of China: International Academic Publishers.

Welford, A. T. (1968). *Fundamentals of skill.* London: Methuen.

Wellman, M. P., Breese, J. S., & Goldman, R. P. (1992). From knowledge bases to decision models. *Knowledge Engineering Review, 7*(1), 35-53.

Wenger, E. (1987). Artificial Intelligence and Tutoring Systems. *Computational and Cognitive Approaches to the Communications of Knowledge*. Morgan Kaufmann Publishers.

Whitehill, B. V., & McDonald, B. A. (1993). Improving learning persistence of military personnel by enhancing motivation in a technical training program. *Simulation and Gaming, 24*(3), 294-313.

Whitesel, H. K., & Fairhead, D. L. (1996). *Requirements for Prediction and Control of Stability and Flooding for Surface Combatants*, NSWCCD-TR-85-96/13, 1986

Whitesel, H. K., & Nemerich, C. P. (1995). *Advanced damage control sensor requirements and technology* (CARDIVNSWC-TR-85-95/01). Bethesda, MD: Naval Surface Warfare Center, Carderock Division.

Wickens, C. D., & Kessel, C. (1980). Processing resource demands of failure detection in dynamic systems. *Journal of Experimental Psychology: Human Perception and Performance, 6*(3), 564-577.

Widman, L. E., Loparo, K. A., & Nielsen, N. R. (1989). *Artificial Intelligence, Simulation, and Modeling*. New York: John Wiley and Sons.

Wilkins, D. C. (1988a). *Apprenticeship learning techniques for knowledge based systems* (STAN-CS-88-1242). Stanford University, Department of Computer Science, 153 pages.

Wilkins, D. C. (1988b). Knowledge base refinement using apprenticeship-learning techniques. In *Proceedings of the 1988 National Conference on Artificial Intelligence* (pp. 646—651). Minneapolis, MN.

Wilkins, D. C., Clancey, W. J., & Buchanan, B. G. (1988). Using and evaluating differential modeling in intelligent tutoring and apprentice learning systems. In Psotka, J., & Massey, D. (Eds.), *Intelligent Tutoring Systems: Lessons Learned* (pp. 257-279). Hillsdale, NJ: Lawrence Erlbaum Associates.

Wilkins, D. C. (1990). Knowledge base refinement as improving an incomplete and incorrect domain theory. In Kodratoff, Y., & Michalski, R. S. (Eds.), *Machine Learning: An Artificial Intelligence Approach: Vol. III* (pp. 493-514). San Mateo, CA: Morgan Kaufmann.

Wilkins, D. C., & Ma, Y. (1994). The refinement of probabilistic rule sets: Sociopathic interactions. *Artificial Intelligence, 70*(1), 1-32.

Wilkins, D. C., (1996). "Inductive Learning for Recursive Heuristic Classification," *Knowledge Acquisition*, in press.

Williams, F.W., Toomey, T.A. and Carhart, H.W., "The ex-USS *Shadwell*, Full-Scale Fire Research and Test Ship," NRL Memorandum Report 6074, 6 October 1987, re-issued September 1992

Williams, F. W. (1995). *Ex-USS Shadwell*: Bibliography, Web Page: http//www.chemistry.nrl.navy.mil/6180/

Wujick, & Rosborough, J. (1992). *Shipboard stability software technical evaluation* (NAVSEA TECHNOTE 070-55W-TN-0003).

Ye, R. L., & Johnson, P. E. (1995). The impact of explanation facilities on user acceptance of expert systems advice. *MIS quarterly, 19*, 157-172.

Zhou, H. H., Silverman, B. G., & Simkol, J. (1989). CLEER: An AI system developed to assist equipment arrangements on warships. *Naval Engineers Journal, 101* (3), 127-137.